Queen's

Master of Management in Artificial Intelligence

MMAI 891

Natural Language Processing

Dr. Stephen W. Thomas

Individual Assignment

March 3, 2019 11:59 PM

Sadman Sakib Hasan (20139537)

Order of files:

Filename	Pages	Comments and/or Instructions
1.0-sh-data-exploration.ipynb		Python notebook for initial data exploration.
2.0-sh-data-preprocessing.ipynb		Python notebook for data preprocessing.
3.0-sh-text-preprocessing.ipynb		Python notebook for text preprocessing.
4.0-sh-text-modelling.ipynb		Python notebook for text modelling using Term Frequency.
5.0-sh-lexicon-model.ipynb		Python notebook to perform lexicon modelling.
6.0-sh-machine-learning-model.ipynb		Python notebook to perform machine learning model using Logistic Regression.

Additional Comments:

Table of Contents

Problem 1 ..... ***3***

Problem 2 ..... ***11***

Problem 3 ..... ***11***

Problem 1

1. a) The chosen dataset for this problem was from an open source database called *Figure Eight*[1]. The name of the dataset is **Judge Emotion About Brands & Products**. The dataset consists of about 9,100 tweets expressing emotions toward a brand and/or product. The dataset contains 3 features: the tweet message, the brand/product the tweet is aimed for and the evaluated emotion of the message.
2. a) **Lexicon approach**: The package used for evaluating the tweets in lexicon approach is called vaderSentiment[2]. VADER is an open source lexicon and rule-based sentiment analysis tool that is built to analyze sentiments expressed in social medias.[3]

   The package uses NLTK under the hood for sentiment analysis. With the use of NLTK in conjunction with VADER it is able to do sentiment analysis on longer texts. It is also great at decomposing paragraphs, articles/reports/publications, or novels into sentence-level analysis[4].

   The package computes a compound score for each sentence computed by summing the valence scores of each word in the lexicon, then adjusting according to the rules and normalizing the score between -1 (extremely negative) to +1 (extremely positive)[5].

   A positive sentiment is given when the compound score is greater than or equal to 0.05, a neutral sentiment is given when the compound score is greater than -0.05 and less than 0.05 and a negative sentiment is given when the compound score is less than or equal to -0.05.

[1] https://www.figure-eight.com/data-for-everyone/
[2] https://github.com/cjhutto/vaderSentiment
[3] https://github.com/cjhutto/vaderSentiment#vader-sentiment-analysis
[4] https://github.com/cjhutto/vaderSentiment#python-code-example
[5] https://github.com/cjhutto/vaderSentiment#about-the-scoring

- **Data exploration:** The first step was to do a basic analysis on all the features in the dataset. This included getting the shape of the dataset, checking for null values, peeking the head and tail of the dataset and finding out the unique human emotions included in the dataset.
- **Data preprocessing:** The next step was to do data processing. This step included label encoding the human emotions for the lexicon approach. I labelled 0 as negative emotion, 1 as neutral emotion and 2 as positive emotion. With this value, I was able to evaluate the accuracy score of the lexicon approach.

b) **ML approach:**

- **Text preprocessing:** Using the NLP techniques learned in the class, I performed text preprocessing on the original tweets in the following order:
    - **Remove all twitter usernames**: None of the twitter usernames were to add any context to the actual message, so I decided to remove all of the twitter handles.
    - **Remove RT symbol**: The symbol RT stands for retweet. I decided to remove all the RT symbols, however, the actual messages of the RT were kept intact as they would be valuable for the sentiment itself.
    - **Remove URL links**: This step included removing all hyperlink values which consisted of a website URL. Again, as the URL would not add any value to perform the sentiment analysis, it was safely removed.
    - **Remove HTML characters:** This included characters like " or & which are used in embedded HTML texts.
    - **Remove unwanted characters:** This included removing all punctuations or unwanted symbols.

- **Remove all numerical values:** This included removing all numerical values.
- **Trim all whitespaces:** This step included trimming all unwanted whitespaces leaving all words delimited by a space only.
- **Normalize all text to American English:** Normalized all words to a standard American grammar to avoid having words both in British and American grammar.
- **Fix spelling:** Fix word spellings using the American dictionary.
- **Remove all proper nouns:** This included dropping all proper nouns such as names of people or countries.
- **Decode Unicode values:** This included decoding all Unicode values to keep everything in standard English.
- **Lemmatization:** This included part-of-speech tagging on each word based on the context it was used and finding the lemmatized version of the word.
- **Case normalization:** Converted all words to lower case to have all words in a standardized manner.
- **Remove stop words:** Remove all the stop words appearing in the American dictionary.
- **Remove rare words:** Remove all words appearing less than 10 times in the entire dataset.
- **Remove frequent words:** Remove all words appearing more than 500 times in the entire dataset.
- **Drop all empty tweet instances:** After all the preprocessing steps, there were 515 instances where the tweet messages were empty. So, I decided to

drop those instances from our data to run the ML model. Similarly, for the Lexicon model I also dropped those row instances to keep the used data for both modelling the same.

- **Represent the word clouds:**

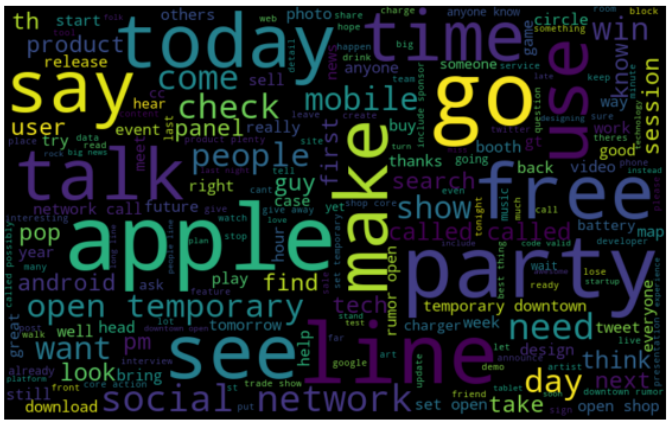

Figure 1. Word clouds representing all **neutral** words

Figure 2. Word clouds representing all **negative** words

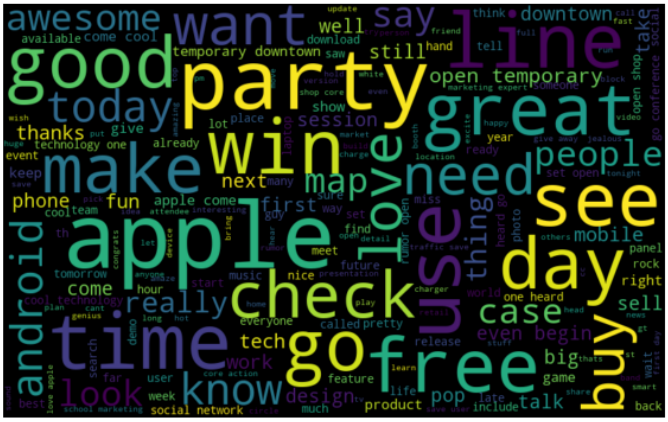

Figure 3. Word clouds representing all **positive** words

- **Text Modelling:**
  - **<u>Term Frequency</u>:** Used term frequency to represent the words in vectorized form.
    - **Hyperparameters for TF:**
      - min_df: 0.02
      - max_features: 14
      - n_grams: [1, 3]
    - Used **CountVectorization** to represent each of the features in vectorized form
    - Represented the words as BOW corupus
    - Used the BOW corpus to run the LDA model to generate topic models
    - **Hyperparameters for LDA:**
      - num_topics=15
      - alpha='asymmetric'
      - eta='auto'
      - passes=20

- iterations=500

- Generated topic models:

	0	1	2	3	4	5	6	7	8	9
Topic 0	go	see	use	apple	line	today	time	party	called	social
Topic 1	open	temporary	apple	win	free	line	social	go	today	party
Topic 2	free	called	party	go	see	line	apple	social	win	time
Topic 3	win	free	open	temporary	go	apple	party	time	see	use
Topic 4	party	go	time	open	line	apple	free	called	social	today
Topic 5	apple	line	party	win	see	use	called	go	open	today
Topic 6	today	win	go	see	line	apple	free	open	called	party
Topic 7	social	today	free	time	use	called	see	go	party	open
Topic 8	time	win	use	free	open	apple	social	go	see	line
Topic 9	use	called	open	today	social	apple	free	line	party	win
Topic 10	use	called	open	today	social	apple	free	line	party	win
Topic 11	temporary	apple	line	go	open	see	today	use	social	free
Topic 12	use	called	open	today	social	apple	free	line	party	win
Topic 13	time	win	use	open	apple	free	social	go	see	line
Topic 14	use	called	open	today	social	apple	free	line	party	win

Figure 4. Generated topic models using LDA algorithm

- **Machine Learning Model:**
  - **<u>Logistic Regression:</u>**
    - Train/test split: 25% test, 75% train
    - Parameters:
      - C = [0.00001, 0.0001, 0.001, 0.01, 1, 10, 100]
      - solver = ["newton-cg", "lbfgs", "sag", "saga"]
      - multi_class = ["ovr", "multinomial", "auto"]
      - max_iter = [100, 200, 400]
      - fit_intercept = [True, False]
    - Cross-validation: 10

3.

a) Confusion matrices:

Predicted	0	1	2	All
Actual				
0	206	154	183	543
1	609	2584	1951	5144
2	255	876	1737	2868
All	1070	3614	3871	8555

Figure 5. Confusion Matrix using **Lexicon** Approach

Predicted	1	2	All
Actual			
0	113	5	118
1	1255	26	1281
2	696	44	740
All	2064	75	2139

Figure 6. Confusion Matrix using **Machine Learning** Approach

b) Accuracy vs Sensitivity vs Specificity

	Accuracy	**Sensitivity**	**Specificity**
Lexicon	0.53	0.50	0.73
ML	0.61	0.35	0.68

Table 1. Accuracy vs Sensitivity vs Specificity for both Lexicon and ML approaches

4.

a) **Lexicon correct, ML incorrect:**

i. **Original tweet:** To kick off #SXSWi @mention is giving away an iPad 2... Just visit the FB page to enter: {link} #SXSW

ii. **Actual prediction:** Positive

iii. **Lexicon prediction:** Positive

iv. **ML prediction:** Neutral

b) **ML correct, Lexicon incorrect:**

i. **Original tweet:** RT @mention Make sure you are donating to the JAPANESE Red Cross for #japan: {link} #sxswcares #sxsw #quake | thank YOU!

ii. **Actual prediction:** Neutral

iii. **Lexicon prediction:** Positive

iv. **ML prediction:** Neutral

c) **Both correct:**

i. **Original tweet:** Next up @mention #sxswi: Your Mom Had An #iPad, Designing For Boomers #SXSW #sxswi

ii. **Actual prediction:** Neutral

iii. **Lexicon approach:** Neutral

iv. **ML approach:** Neutral

d) **Both incorrect:**

i. **Original tweet:** so the iPad will be available while I'm in Austin for #sxsw -- this is major #GeekDilemma

ii. **Actual prediction:** Positive

iii. **Lexicon prediction:** Neutral

iv. **ML prediction:** Neutral

5. The ML approach performed better than the basic lexicon approach as expected, however, it did not perform significantly well (achieved an accuracy score of 61% vs 53% for lexicon approach). This is mostly because of the actual human sentiment prediction of some tweets were not correctly labelled and as a result our ML algorithm predicted a lot of false positives for negative tweets as either neutral tweets or positive tweets. Figure 1 to 3 depicts

the word cloud for each sentiment words, and it is noticeable how a lot of the words overlap between negative and neutral sentiment causing our ML algorithm to predict a lot of false positives. A better dataset with more accurate initial predictions would have been better to help our ML algorithm learn better but regardless it was still able to predict the sentiments correctly than the lexicon approach.

Problem 2

Done as part of pull request: https://github.com/stepthom/text_mining_resources/pull/5

Problem 3

1. I am currently working as a Software Engineer at IBM. The project my team working on is an application to enhance software development experience in Hybrid Cloud, known as Microclimate. Our customers are software developers themselves and the business model revolves around enhancing their development experience for both in cloud and on-premise platforms. We provide an end-to-end development solution starting from writing the code, testing it, managing the code directly from cloud and finally deploying them using pipeline models. Our software product is free to use but the free version is only available for local development. The main source of our revenue is generated from the IBM Cloud Private user base who can select to buy Microclimate as a service using the cloud cluster instance.
2.
    - **Opportunity #1:**
        a. A simple slack chat-bot using rule-based learning.

b. This chat bot will be used to help answer simple questions related to our product posted by the users in the slack channel and will be built and maintained by the Microclimate team.

c. Our users will primarily be the internal users in the IBM organization slack or the public users in our external channel. The dialog system will be used frequently every day as there are a lot of unanswered questions currently.

d. It would significantly improve the existing process as the current process is just the regular employees trying to answer each question asked by the user. With this dialog system, the user can benefit from quick solution to their problems and as well as the regular employees don't need to keep an eye on the channel all the time anymore.

e. **John:** I am having trouble setting up NFS for my Microclimate service on ICP. Help anybody?

Agent: Hello John, please follow the instructions mentioned at https://www.ibm.com/help/microclimate/icp/nfs for more information.

f. Since it is a rule-based dialog system, it won't be able to answer all questions accurately for the users. If the user has a specific question about a specific problem, it will fail to answer correctly to the user. Whereas if there is a human answering to the user, he/she might be able to provide exact troubleshooting steps for the specific problem.

- **Opportunity #2:**

a. A corpus-based chat-bot using information retrieval modelling plugged into the Microclimate web application in cloud for users to get help right away if they need any help.

b. This chat bot will be able to answer questions related to our product in a one-to-one chat-based manner. This dialog system can be developed and maintained by the Microclimate team or can be outsourced to a machine learning team in IBM.

c. Our users will be the ones who have installed Microclimate on their local development machine or using Microclimate as a service in the private cloud instance. The dialog system will be available for use as part of the Microclimate product.

a. Currently, we have zero attention to real time help our customers want directly when using the software product. So, having such a chat-bot plugged in directly to our software will be a great addition to help our customers quickly solve a problem.

d. **Agent:** Hello, welcome to our enhanced Microclimate chat service. What may I help you with today?

John: Hi, how can I upload an existing Node.js project to my cloud?

Agent: Hi John, you can import an existing project by simply clicking the import button on the bottom left and selecting the type of project import you want to perform, e.g from GitHub, from zip file or from local folder.

e. Some areas of concerns using this model could be related to the questions user asks may not be related to Microclimate at all and since our corpus will contain only Microclimate related answers, our machine learning model will not know how to answer such questions.

- **Opportunity #3:**
  a. A task-oriented dialog box to help the customers solve a problem with real-time step by step instructions.
  b. This dialog system will be used to help answer questions related to our product by letting the users follow a set of tasks in a particular order. Users will have real time interactions with the dialog box and follow each instruction to overcome the problem. This dialog system can be developed and maintained by the Microclimate team or can be outsourced to a machine learning team in IBM.
  c. Our users will be the ones who have installed Microclimate on their local development machine or using Microclimate as a service in the private cloud instance. The dialog system will be available for use as part of the Microclimate product.
  d. Currently, we have zero attention to real time help our customers want directly when using the software product. So, having such a task-oriented dialog system plugged in directly to our software will be a great addition to help our customers quickly solve a problem.
  e. **Agent:** Hello, welcome to our enhanced Microclimate chat service. What may I help you with today?

     John: Hi, how can I upload an existing Node.js project to my cloud?

     Agent: Hi John, let's get started with an interactive step. First click on the import button on the bottom left. After you have done that enter done in the chat. If that step failed, please enter a description on how it failed for me to help you investigate the failure further.

     John: Done.

Agent: Great! Next, select the type of import you want to do. Enter the type of import you want to do on the chat for specific help on that import type, e.g GitHub or Zip or Local. Enter skip to go the next step.

f. One of the main concerns is that since our task-oriented dialog system will pretty much now guide the user to achieve a goal, the goal could be infinitely long, or the user could just mess around with the agent by infinitely looping around the same tasks over and over again. Another potential concern could be if the user hits a problem, but our agent does not have the right solution to it, it will be stuck at providing the wrong solution to the user all the time.

3.

a. Opportunity #2 will be the best case for the Microclimate product to have right now. It will provide an intelligent touch to the product to acquire product feedback as well. The return of invest for having such dialog system for this product will be significant for number of reasons:
    - Customer problems will be almost always solved right away; making our customers extremely happy and attract more people to use it.
    - There will be none to minimal amount of extra human work needed to track customer questions as the chat bot will be able to now solve almost all in an automated manner.
    - We can deduce the area of the product that needs the most attention (code or documentation wise), depending on which area of the product hits the most questions. It will make the project manager's life much easier to prioritize work that needs to be done. If an area of the product is being hit

with a lot of questions, as a project manager, I will make sure to allocate more resources to that area to fix it as soon as possible.

b. The development timeline for this chatbot could be anywhere between 3-4 months with proper testing and documentation if the correct resources are allocated, i.e if a team with actual machine learning engineers are working on it. For example, if the Microclimate team decides to not outsource this project and decide to build it themselves, it will not be that great as there is no machine learning engineers in the team currently and it will take a longer amount of time for them. However, if the product is outsourced to an internal IBM machine learning team for a discount price and simply plugged into the Microclimate framework, it will be much quicker and more efficient, and more importantly the chatbot will be working with an adequate level of functionality.