

**MMAI 845**

**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** |
|  |  |  |
|  |  |  |

**Additional Comments:**

Table of Contents

[Problem 1 3](#_Toc2352293)

# **Problem 1**

1. a) The dataset chosen for this problem was from an open source database called Figure Eight[[1]](#footnote-2). 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 a product. The dataset contains 3 features: the tweet message, the brand/product the tweet is aimed for and the evaluated emotion of the message. The dataset is reliable with the human mapping of the initial emotion expressed towards the tweet message.
2. a) **Lexicon approach**: The package used for evaluating the tweets in lexicon approach is called vaderSentiment[[2]](#footnote-3). VADER is an open source lexicon and rule-based sentiment analysis tool that is built to analyze sentiments expressed in social medias.[[3]](#footnote-4)

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]](#footnote-5).

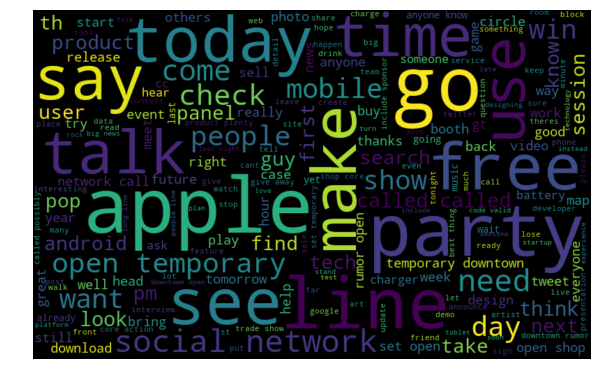
The package computes a compound score for each sentence computed by summing the valence scores of each word in the lexicon, then adjusting according to the rules and normalizing the score between -1 (extremely negative) to +1 (extremely positive)[[5]](#footnote-6).

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.

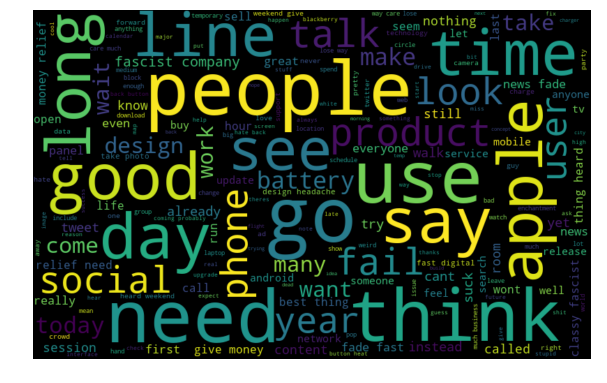
* **Data exploration:** The first step was to do a basic analysis on all the features in the dataset. This included getting the shape of the dataset, checking for null values, peeking the head and tail of the dataset and finding out the unique human emotions included in the dataset.
* **Data preprocessing:** 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, we were able to evaluate the accuracy score of the lexicon approach.

b) **ML approach:**

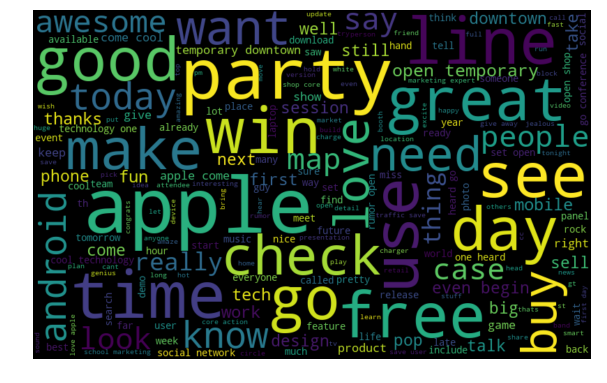
* **Text preprocessing:** Using the NLP techniques we learned in the class, I performed text preprocessing on the original tweets in the following order:
  + **Remove all twitter usernames**: None of the twitter usernames were to add any context to the actual message, so I decided to remove all of the 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 &quot; or &ampt; 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 spelling of words 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.
* **Represent the word clouds:**



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

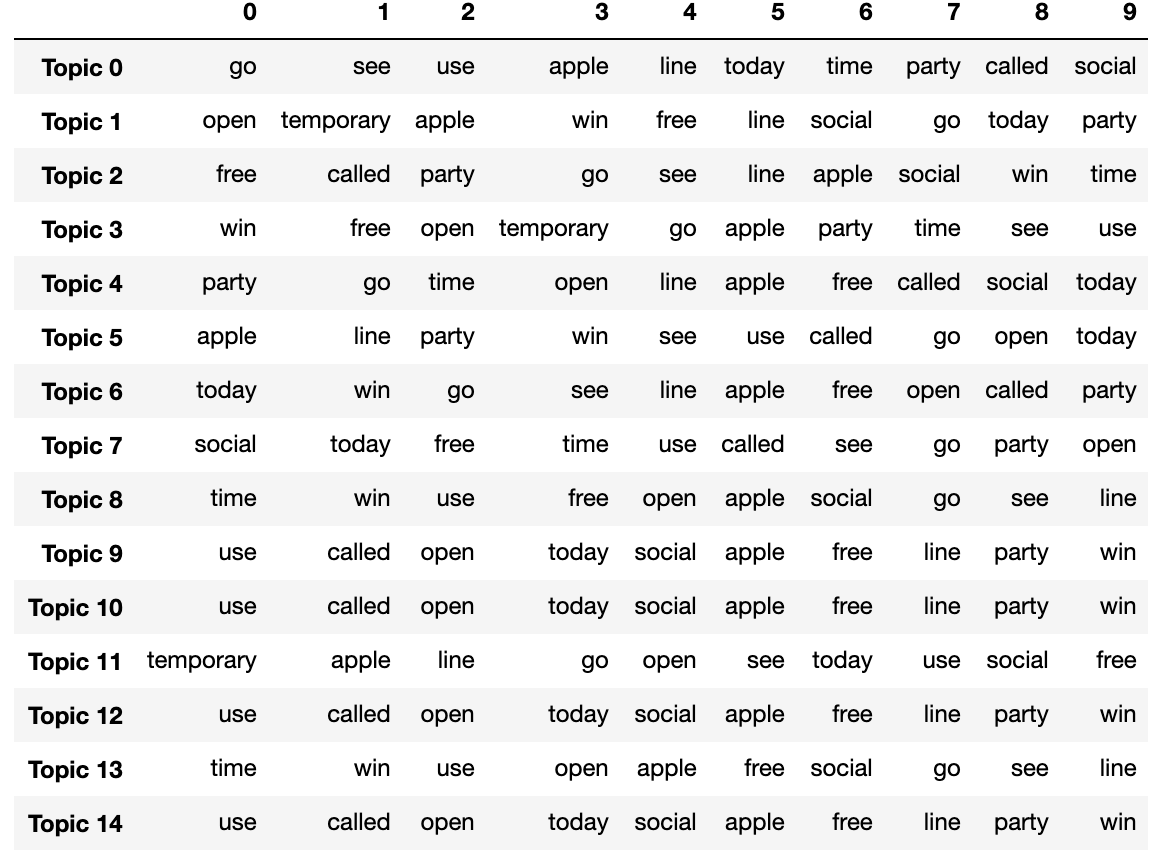


**Figure 2.** Word clouds representing all **negative** words



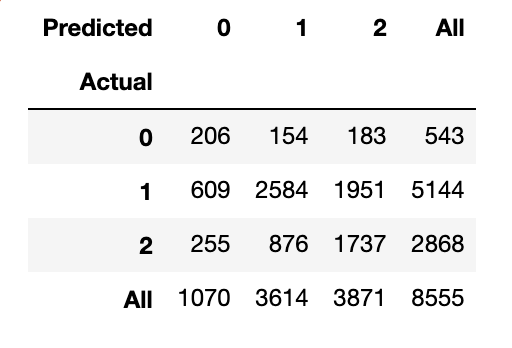
**Figure 3.** Word clouds representing all **positive** words

* **Text Modelling:**
  + **Term Frequency:** 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:

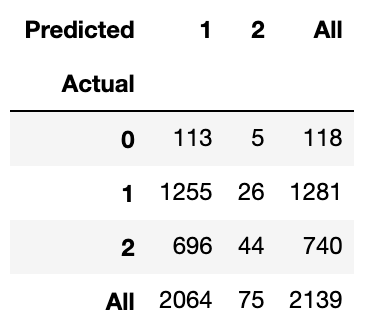


**Figure 4.** Generated topic models using LDA algorithm

* **Machine Learning Model:**
  + **Logistic Regression:**
    - 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
  1. Confusion matrices:



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



**Figure 6.** Confusion Matrix using Lexicon Approach



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

1. 1. **Lexicon correct, ML incorrect:**
      1. **Original tweet:** To kick off #SXSWi @mention is giving away an iPad 2... Just visit the FB page to enter: {link} #SXSW
      2. **Actual prediction:** Positive
      3. **Lexicon prediction:** Positive
      4. **ML prediction:** Neutral
   2. **ML correct, Lexicon incorrect:**
      1. **Original tweet:** RT @mention Make sure you are donating to the JAPANESE Red Cross for #japan: {link} #sxswcares #sxsw #quake | thank YOU!
      2. **Actual prediction:** Neutral
      3. **Lexicon prediction:** Positive
      4. **ML prediction:** Neutral
   3. **Both correct:**
      1. **Original tweet:** Next up @mention #sxswi: Your Mom Had An #iPad, Designing For Boomers #SXSW #sxswi
      2. **Actual prediction:** Neutral
      3. **Lexicon approach:** Neutral
      4. **ML approach:** Neutral
   4. **Both incorrect:**
      1. **Original tweet:** so the iPad will be available while I'm in Austin for #sxsw -- this is major #GeekDilemma
      2. **Actual prediction:** Positive
      3. **Lexicon prediction:** Neutral
      4. **ML prediction:** Neutral
2. 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%). 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 some the sentiment correctly than the lexicon approach.

# **Problem 2**

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

1. <https://www.figure-eight.com/data-for-everyone/> [↑](#footnote-ref-2)
2. <https://github.com/cjhutto/vaderSentiment> [↑](#footnote-ref-3)
3. <https://github.com/cjhutto/vaderSentiment#vader-sentiment-analysis> [↑](#footnote-ref-4)
4. <https://github.com/cjhutto/vaderSentiment#python-code-example> [↑](#footnote-ref-5)
5. <https://github.com/cjhutto/vaderSentiment#about-the-scoring> [↑](#footnote-ref-6)