Social Media Analytics Project

By: - Tushar Pant, MsBA 2021, DAT-9031-BOS1

Introduction

Nowadays many government institutions and companies need to know their customers' feedback and comment on social media such as Facebook. Sentiment analysis is one of the best modern branches of machine learning, which is mainly used to analyze the data to know one's own idea, nowadays it is used by many companies to their own feedback from customers.

Why should we use sentiment analysis?

- **Invaluable Marketing:**
  Using sentiment analysis companies and product owners use can use sentiment analysis to know the demand and supply of their products through comments and feedback from the customers.
- **Identifying key emotional triggers:**
  In psychology and other medical treatment institutions, sentiment analysis can be used to detect whether the individuals' emotion is normal or abnormal, and based on the data record they can decide person health.
- **Politics:**
  In the political field, candidates to be elected can use sentiment analysis to predict their political status, to measure people's acceptance. It can also be used to predict election results for electoral board commissions.
- **Education:**
  Universities and other higher institutes like colleges can use sentiment analysis to know their student's feedback and comment, therefore they can take consideration to revise or improve their education curriculum.

Application

- We used to download the Facebook comment dataset from the Kaggle website which is the best dataset provider.
- For the code we already used "*fb_sentiment.txt*" for analysis of kindle amazon Facebook comment, you can use your own Facebook comment using this code to analyze your own comments or create a file in text format and try it for simplification.

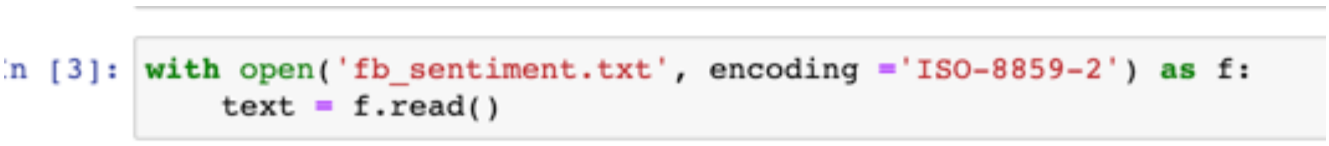
We follow these major steps in our program:

- Downloading(fetching) Facebook comment from Kaggle site and save it as text format.
- Preprocessing the data through SkLearn and nltk libraries. We first tokenize the data and then after tokenizing we stemize and lemmatize.

- Parse the comments using Vader library. Classify each comment as **positive, negative, or neutral**.

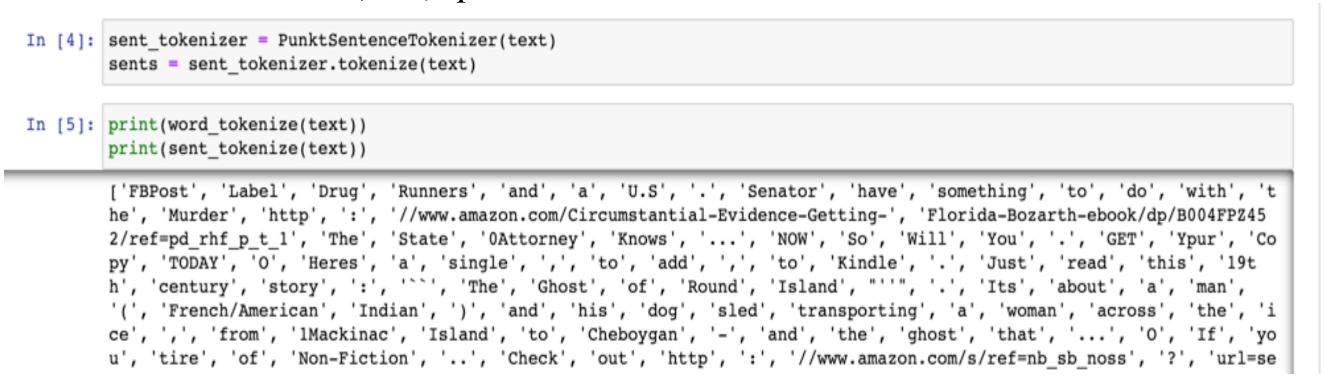
Now, let us try to understand the piece of code:

- First, we open a file named kindle which is downloaded from Kaggle site and saved in local disk.



```
n [3]: with open('fb_sentiment.txt', encoding ='ISO-8859-2') as f:
           text = f.read()
```

- After we open a file, we preprocess the text through tokenize, stemize and then lemmatize:
- Tokenize the text, i.e., split words from text.



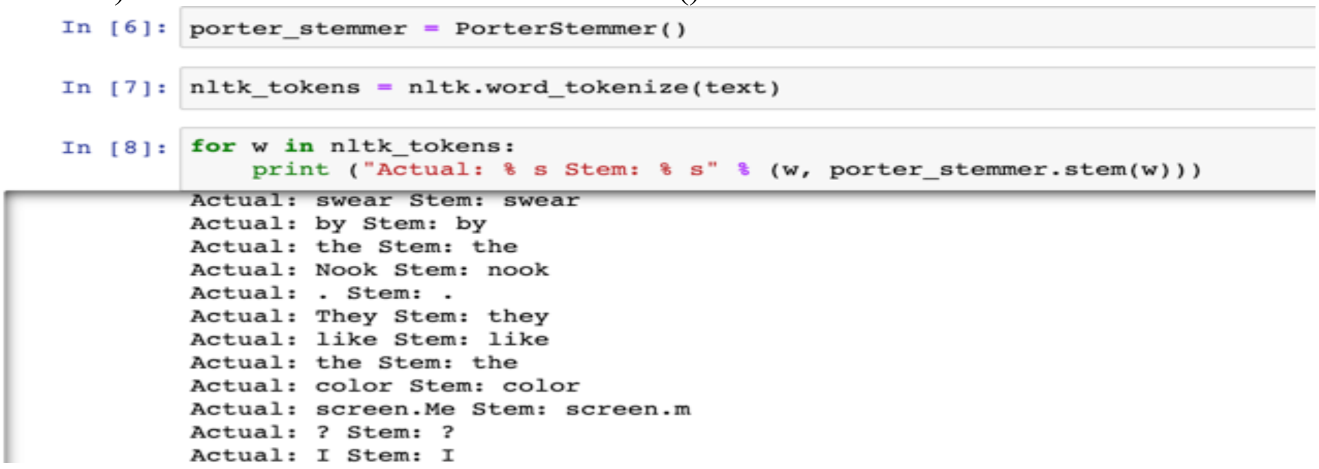
```
In [4]: sent_tokenizer = PunktSentenceTokenizer(text)
        sents = sent_tokenizer.tokenize(text)

In [5]: print(word_tokenize(text))
        print(sent_tokenize(text))

        ['FBPost', 'Label', 'Drug', 'Runners', 'and', 'a', 'U.S', '.', 'Senator', 'have', 'something', 'to', 'do', 'with', 't
        he', 'Murder', 'http', ':', '//www.amazon.com/Circumstantial-Evidence-Getting-', 'Florida-Bozarth-ebook/dp/B004FPZ45
        2/ref=pd_rhf_p_t_1', 'The', 'State', '0Attorney', 'Knows', '...', 'NOW', 'So', 'Will', 'You', '.', 'GET', 'Ypur', 'Co
        py', 'TODAY', 'O', 'Heres', 'a', 'single', ',', 'to', 'add', ',', 'to', 'Kindle', '.', 'Just', 'read', 'this', '19t
        h', 'century', 'story', ':', '``', 'The', 'Ghost', 'of', 'Round', 'Island', "''", '.', 'Its', 'about', 'a', 'man',
        '(', 'French/American', 'Indian', ')', 'and', 'his', 'dog', 'sled', 'transporting', 'a', 'woman', 'across', 'the', 'i
        ce', ',', 'from', '1Mackinac', 'Island', 'to', 'Cheboygan', '-', 'and', 'the', 'ghost', 'that', '...', 'O', 'If', 'yo
        u', 'tire', 'of', 'Non-Fiction', '..', 'Check', 'out', 'http', ':', '//www.amazon.com/s/ref=nb_sb_noss', '?', 'url=se
```

- Stemize and lemmatize the text for normalization of the text:

1) For stemize we use Porter Stemmer () function:

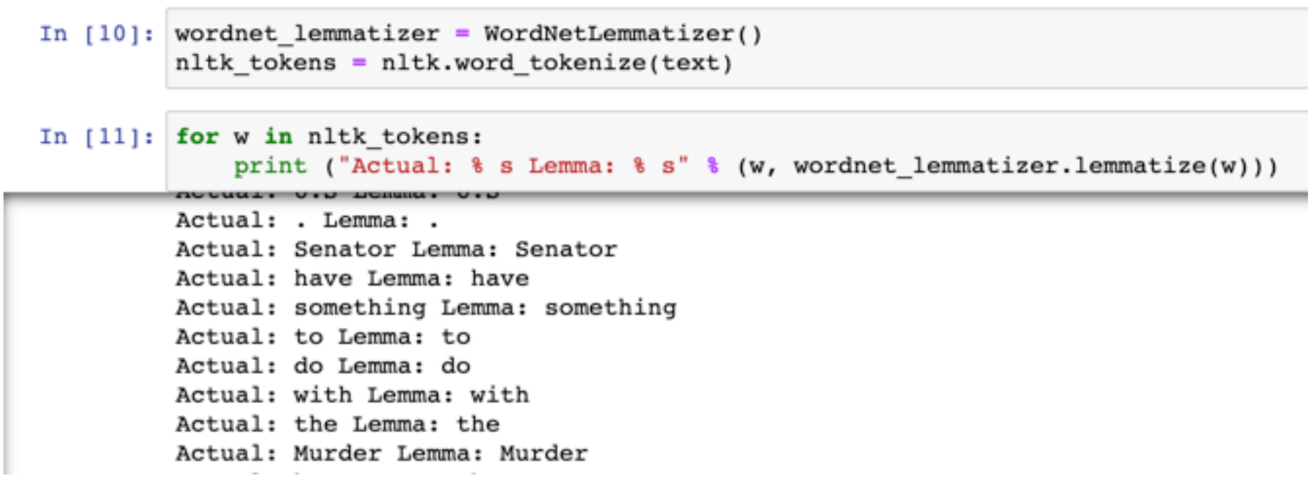


```
In [6]: porter_stemmer = PorterStemmer()

In [7]: nltk_tokens = nltk.word_tokenize(text)

In [8]: for w in nltk_tokens:
            print ("Actual: % s Stem: % s" % (w, porter_stemmer.stem(w)))
        Actual: swear Stem: swear
        Actual: by Stem: by
        Actual: the Stem: the
        Actual: Nook Stem: nook
        Actual: . Stem: .
        Actual: They Stem: they
        Actual: like Stem: like
        Actual: the Stem: the
        Actual: color Stem: color
        Actual: screen.Me Stem: screen.m
        Actual: ? Stem: ?
        Actual: I Stem: I
```

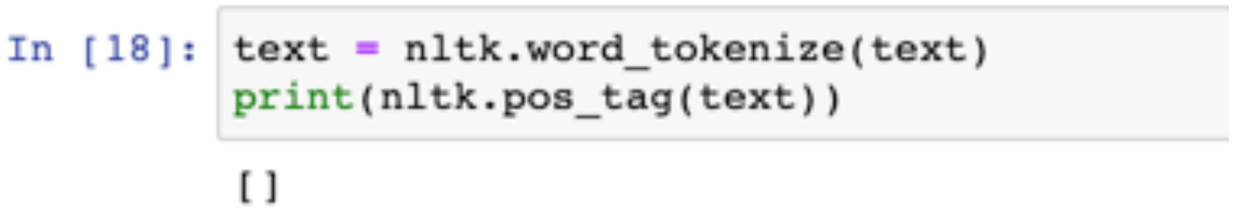
2) For lemmatize we use WordNetLemmatizer () function:



```
In [10]: wordnet_lemmatizer = WordNetLemmatizer()
         nltk_tokens = nltk.word_tokenize(text)

In [11]: for w in nltk_tokens:
             print ("Actual: % s Lemma: % s" % (w, wordnet_lemmatizer.lemmatize(w)))
         Actual: U.S Lemma: U.S
         Actual: . Lemma: .
         Actual: Senator Lemma: Senator
         Actual: have Lemma: have
         Actual: something Lemma: something
         Actual: to Lemma: to
         Actual: do Lemma: do
         Actual: with Lemma: with
         Actual: the Lemma: the
         Actual: Murder Lemma: Murder
```

- **POS** (part of speech) tagging of the tokens and select only significant features/tokens like adjectives, adverbs, and verbs, etc.
- Pass the tokens to a sentiment intensity analyzer which classifies the Facebook comments as positive, negative, or neutral.



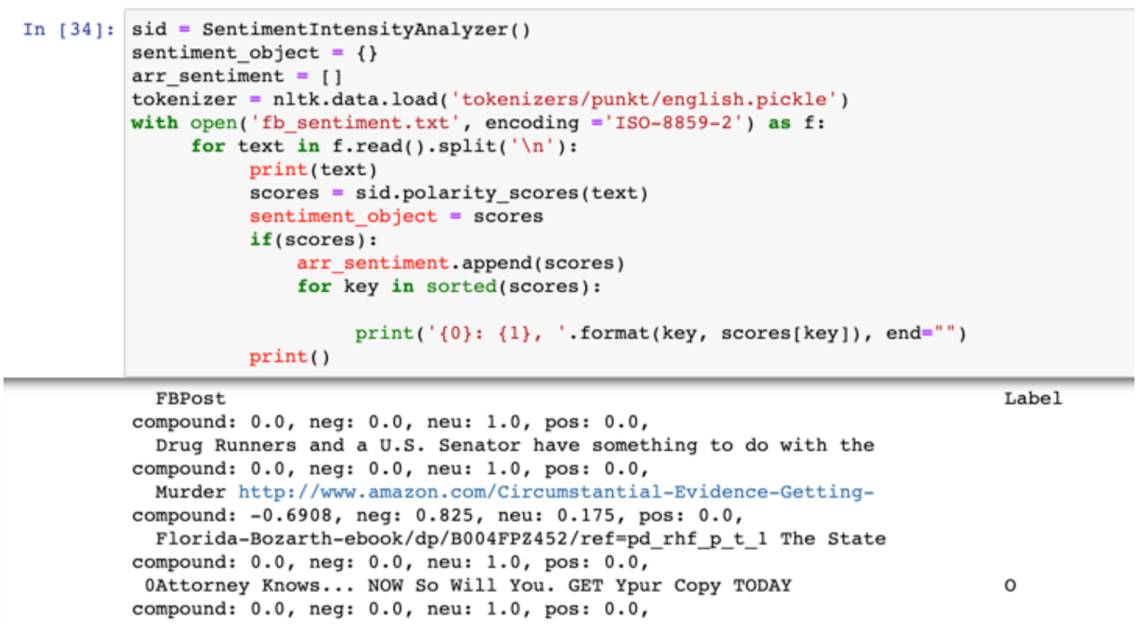
```
In [18]: text = nltk.word_tokenize(text)
         print(nltk.pos_tag(text))

         []
```

How Vader sentiment analyzer works:

- VADER uses a combination of A sentiment lexicon which is a list of lexical features (e.g., words) which are generally labeled according to their semantic orientation as either positive or negative.
- Sentiment analyzer not only talks about the Positivity and Negativity score but also tells us about how positive or negative a sentiment is.
- Then, we used the polarity scores () method to obtain the polarity indices for the given sentence.

- Then, we build the comment intensity and polarity as:



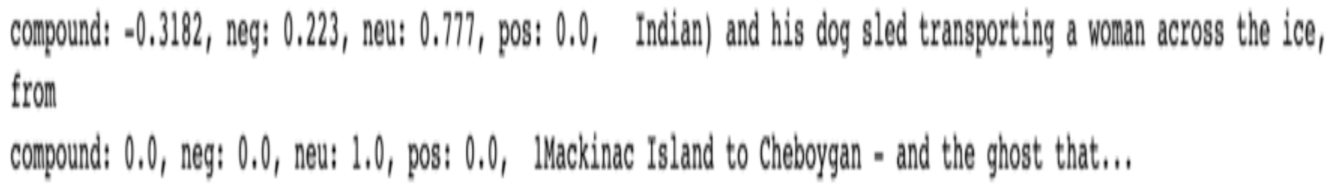
```
In [34]: sid = SentimentIntensityAnalyzer()
         sentiment_object = {}
         arr_sentiment = []
         tokenizer = nltk.data.load('tokenizers/punkt/english.pickle')
         with open('fb_sentiment.txt', encoding ='ISO-8859-2') as f:
             for text in f.read().split('\n'):
                 print(text)
                 scores = sid.polarity_scores(text)
                 sentiment_object = scores
                 if(scores):
                     arr_sentiment.append(scores)
                     for key in sorted(scores):

                         print('{0}: {1}, '.format(key, scores[key]), end="")
                 print()
```

```
           FBPost                                                                  Label
         compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,
           Drug Runners and a U.S. Senator have something to do with the
         compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,
           Murder http://www.amazon.com/Circumstantial-Evidence-Getting-
         compound: -0.6908, neg: 0.825, neu: 0.175, pos: 0.0,
           Florida-Bozarth-ebook/dp/B004FPZ452/ref=pd_rhf_p_t_1 The State
         compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,
          0Attorney Knows... NOW So Will You. GET Ypur Copy TODAY                 O
         compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,
```

- Python arrays are a data structure like lists. They contain several objects that can be of different data types. In addition, Python arrays can be iterated and have several built-in functions to handle them

Let us to understand what the sentiment code is and how VADER performs on the output of the above code:



```
compound: -0.3182, neg: 0.223, neu: 0.777, pos: 0.0,   Indian) and his dog sled transporting a woman across the ice,
from
compound: 0.0, neg: 0.0, neu: 1.0, pos: 0.0,  1Mackinac Island to Cheboygan - and the ghost that...
```

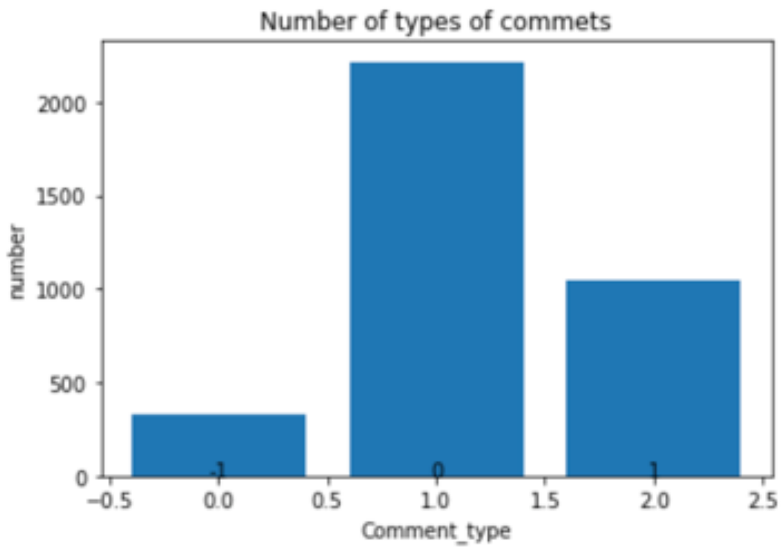
- The Positive(pos), Negative(neg) and Neutral(neu) scores represent the proportion of text that falls in these categories.
- This means our sentence 1 was rated as 0% Positive, 77.7% Neutral and 22.3% Negative. Hence all these should add up to 1 & similarly for the second statement which more *neutral in nature*.
- The Compound score is a metric that calculates the sum of all the lexicon ratings which have been normalized between -1(extreme negative) and +1 (extreme positive).
- Visual representation of the sentimental correlation within the selected dataset:

```
In [44]: arr
Out[44]: {'postive': 1048, 'negative': 326, 'nuetral': 2216}
```

```
In [65]: def addlabels(x,y):
             for i in range(len(x)):
                 plt.text(i,y[i],y[i], ha = 'center')

         x = ['postive', 'nuetral', 'negative']
         y = [-1, 0, 1]

         plt.bar(range(3), [negative_sum, neutral_sum, pos_sum]);
         addlabels(x, y)
         plt.title('Number of types of commets');
         plt.xlabel('Comment_type');
         plt.ylabel('number');
```



Finally, sentiment scores of comments with compound & polarity values are returned.

Inspiration

Possible uses for this dataset could include:

- Sentiment analysis in a variety of forms categorizing Facebook texts based on their comments and statistics.
- Training ML algorithms to generate their own sentimental comments.
- Analyzing what factors affect how popular a Facebook post or comment or any activity, will be.

References:

- Cambria, E., Das, D., Bandyopadhyay, S., & Feraco, A. (Eds.). (2017). A practical guide to sentiment analysis (pp. 1-196). Cham, Switzerland: Springer International Publishing.
- Ahmad, K. (Ed.). (2011). Affective computing and sentiment analysis: Emotion, metaphor and terminology (Vol. 45). Springer Science & Business Media.
- Wu, C., Le Vine, S., Bengel, E., Czerwinski, J., & Polak, J. (2021). Sentiment analysis of popular-music references to automobiles, 1950s to 2010s. Transportation, 1-38.

Appendix:

- **https://github.com/tusharpant93/Sentimental-analysis-and-EDA**
- **http://localhost:8888/notebooks/Desktop/MsBA%20HULT%20BST/Social%20Media%20Analytics/Final%20Project%20SMA.ipynb**