Sakshi Patil

SENTIMENT ANALYSIS

Pre-processing the Tweets

- Letter casing: Converting all letters to either upper case or lower case.
- Tokenizing: Turning the tweets into tokens. Tokens are words separated by spaces in a text.
- Noise removal: Eliminating unwanted characters, such as HTML tags, punctuation marks, special characters, white spaces etc.
- Stopword removal: Some words do not contribute much to the machine learning model, so it's good to remove them. A list of stopwords can be defined by the nltk library, or it can be business-specific.
- Normalization: Normalization generally refers to a series of related tasks meant to put all text on the same level. Converting text to lower case, removing special characters, and removing stopwords will remove basic inconsistencies. Normalization improves text matching.
- Stemming: Eliminating affixes (circumfixes, suffixes, prefixes, infixes) from a word in order to obtain a word stem. Porter Stemmer is the most widely used technique because it is very fast. Generally, stemming chops off end of the word, and mostly it works fine.

Create a dictionary where the keys are a (word, label) tuple and the values are the corresponding frequency. The labels I used here are 1 for positive and 0 for negative.

lookup() function takes in the freqs dictionary, a word, and a label (1 or 0) and returns the number of times that word and label tuple appears in the collection of tweets.

Count tweets() that takes a list of tweets as input, cleans all of them, and returns a dictionary.

Sentiment Analysis using Naive Bayes

I have used Naive bayes algorithm for sentiment analysis.

- a probability for each class is created. $P(D_{pos})$ is the probability that the document is positive. $P(D)$ is the probability $P(D_{neg})$ that the document is negative.

$$P(D_{neg}) = P(D_{neg}) / D$$

$$P(D_{pos})= P(D_{pos}) /D$$

Where D is the total number of documents, or tweets in this case, D_{pos} is the total number of positive tweets and D_{neg} is the total number of negative tweets.

Prior and Logprior

The prior probability represents the underlying probability in the target population that a tweet is positive versus negative.
The prior is the ratio of the probabilities $P(D_{pos})/P(D_{neg})$. We can take the log of the prior to calculate logprior.

$$logprior=log(P(D_{pos})/P(D_{neg}))=log(D_{pos} / D_{neg})$$

Positive and Negative Probability of a Word

To compute the positive probability and the negative probability for a specific word in the vocabulary, we'll use the following inputs:

- $freq_{pos}$ and $freq_{neg}$ are the frequencies of that specific word in the positive or negative class. In other words, the positive frequency of a word is the number of times the word is counted with the label of 1.
- N_{pos} and N_{neg} are the total number of positive and negative words for all documents (for all tweets), respectively.
- V is the number of unique words in the entire set of documents, for all classes, whether positive or negative.

We use this formula to compute the positive and negative probability for a specific word:

$$P(W_{pos})= freq_{pos} +1/ N_{pos}+V$$

$$P(W_{neg})=freq_{neg} +1/N_{neg}+V$$

we add the "+1" in the numerator for additive smoothing

Log likelihood

To compute the loglikelihood of that very same word, we can use following equation:

$$loglikelihood=log(P(W_{pos})/ P(W_{neg}))$$

- Using count_tweets() function, you can compute a dictionary called freqs that contains all the frequencies.
- In this freqs dictionary, the key is the tuple (word, label) and the value is the number of times it has appeared.

naive_bayes_predict() function used to make predictions on tweets.

- The function takes in the tweet, logprior, loglikelihood.
- It returns the probability that the tweet belongs to the positive or negative class.
- For each tweet, it sums up loglikelihoods of each word in the tweet.
- Also adds the logprior to this sum to get the predicted sentiment of that tweet.

$$p = logprior + \sum (loglikelihood_i)$$

The function naive_bayes_predict() returns the value of prediction.

If the value of p is greater than zero then it is a positive sentiment else it is a negative sentiment.

Testing naive bayes

test_naive_bayes() takes logprior and loglikelihood and test data and returns the accuracy.