IST 664

Natural Language Processing

Homework 3

Dhwani Rekhang Gandhi

dgandhi@syr.edu

Table of Contents

1: Creating Word Features:

I first created a list of words called word_features which collected all the words from the sentence_polarity corpus that I imported. It consisted of sentences which were labeled as ‘pos’ or ‘neg’. I used this labelled data to train the models with different features so that it can be later used to classify the unlablled review contents of the Amazon Product Data which we have cleaned for analysis previously. I then extracted the sentences from the sentence_polarity corpus into a list called document. I randomly shuffled the sentences in the document. I then extracted the words from this document and stored it in the list called all_words_list. I then selected the most frequent 3000 words and stored them in a list called word_features.

2: Classification Model 1 using NaiveBayesClassifier:

I defined a function document_features which is a Bag of Words. This function returned features which were in a form of a dictionary which consisted of the word as a key and True or False as a value that indicated the presence of the word. I then created a featuresets list which consisted of the features along with the category which they belong to using the document_features function. I then divided the featuresets into training and testing sets in a ratio equal to 91:9. I then used the nltk.NaiveBayesClassifier for training the model. I trained the training set and then tested it on the test set to find the accuracy. It resulted in an accuracy of 77.9%

Now the task was to apply my unlabelled reviews on the model trained. I imported my data and tokenized it based on the sentences because in the end we need two lists of positive and negative sentences. These sentences were stored in a list called tokentext. I ran a loop iterating through the tokentext list. Each sentence was first word tokenized because our word_features function accepts a word tokenized list. I then applied filters to each of the sentence which is now split into words. I first converted into lower case, then selected only words consisting of alphabets and then removed any kind of stop words present. I did so to improve the processing speed of the model. I then used the classifier I had created to classify my sentence. The output of the classify function is one of the labels i.e ‘pos’ or ‘neg’. The output was stored in a variable ‘category’. If it was ‘pos’ then it would be appended to the ‘pos_sent’ list else, it would be appended to the ‘neg_sent’ list. I then checked for the length of the two lists and found the negative list to be more than the positive list.

3: Classification Model 2 using NaiveBayesClassifier:

The second feature I used was detecting negation words. I first created a list of many negation words like 'not', 'no', 'never', 'hardly' etc. I then defined a function called NOT_features. In this function I went through the document words in order of adding the word features, but if the word follows a negation word, I changed the feature to a negated word. I then created a NOT_featuresets from the document list created and used the NOT_features function defined. I then divided the NOT_featuresets into training and testing sets in a similar ratio as the previous step. I then trained classifier1 using the NOT_featuresets using the training set and then tested it on the testing set. I got an accuracy of 79.4%.

Similar to the previous model, I ran a loop iterating through the tokentext list. Each sentence was first word tokenized. I then applied filters that converted the words into lower case, then selected only words consisting of alphabets and then removed any kind of stop words present. I then used the classifier1 I had created to classify my sentence. The output was stored in a variable 'category'. If it was 'pos' then it would be appended to the 'pos_sent1' list else, it would be appended to the 'neg_sent1' list. I then checked for the length of the two lists and found the negative list to be more than the positive list.

4: Classification Model 3 using SklearnClassifier:

I then decided to explore one of the sklearn classifiers to check for accuracy and to classify the model. I imported the BernoulliNB from sklearn.naive_bayes. I then created a classifier2 and classifier3 using different training and testing sets using the previous two features defined. Classifier2 showed an accuracy of 77.7% on using the first feature set which used the document_features function in Classification Model 1 and classifier3 showed an accuracy of 78.2% on using the second feature set which used the NOT_features function in Classification Model 3. Since the accuracy on using the NOT_featuresets was higher I used that for the Classification task.

Similar to the previous model, I ran a loop iterating through the tokentext list. Each sentence was first word tokenized. I then applied filters that converted the words into lower case, then selected only words consisting of alphabets and then removed any kind of stop words present. I then used the classifier3 I had created to classify my sentence. The output was stored in a variable 'category'. If it was 'pos' then it would be appended to the 'pos_sent2' list else, it would be appended to the 'neg_sent2' list. I then checked for the length of the two lists and found the negative list to be more than the positive list.

5: Summary Table:

Method	Accuracy	Positive Sentences	Negative Sentences
Naive-Bayes 'document_features'	77.9%	440,573	700,069
Naive-Bayes 'NOT_features'	79.4%	447,447	693195
BernoulliNB 'document_features'	77.7%	-	-
BernoulliNB 'NOT_features'	78.2%	431938	708704

After creating 3 lists of positive and negative sentences each I decided to use pos_sent1 and neg_sent1 since the classifier used to classify the sentences had the highest accuracy. Below are few examples of the sentences classified as positive and negative by the Classification Model 2.

Positive Sentences	Negative Sentences
'Beautiful vibrant color.', 'I bought several more colors!', 'Nice and puffy tutu skirt.', 'Bought this for my niece as part of her fairy outfit.', 'I bought this for her for Christmas and she never wanted to take it off.'	"Can't recommend.", 'Fits great and easy to clean!', 'Never GOT this item - but gave a 1 STAR because the replies from the SUPPLIER was GREAT.They tried to send the item more than once.My $ was refunded in a timely manner too.It was a shame I never got it for my daughter - it would of looked great with her OUTFIT for Dr. Seuss WEEK at school.Most original.Maybe next time.', 'I would recommend this for girls under 10 yr. old.', 'It will be too short and small for older girls.'