
Please summarize your method:

1) **Software Used: -** PHP (as script)
2) **Features: -** Two bags namely sports.txt and politics.txt of list of all the words present in the training data based on labels 'sports' and 'politics'. The special characters like {(,),:,?...} are removed from data and only the tweets of length more than 2 as anything less than that may provide only noise. The #tagged words are preserved as later they are given 100 times more weightage than a normal word.

Please describe the algorithms in details:

1) **Pre-processing step: -** Nothing specific here the file training.txt is used as it is.

2) **Training algorithm: -**
Input-format: - < tweet_id lable tweet >
Output-format: - 2 files containing all the words based on the algorithm described below.

ALGO:
a) Read the training data from the file.

b) Read the tweet id and tweet label and tweet, use only tweets more than length 2 as smaller tweets provide more noise (assumption). Also, words of length more than 4 are used as they are more significant (assumption), links are ignored and not used at all.

c) Replace all the punctuations from the tweets using regular expression. But keep the #tagged words as it is as they have more weightage.

d) If label was "Sports" put in the bag for sports (sports.txt) all the separate words of the tweet.

e) If label was "Politics" put in the bag for politics (politics.txt) all the separate words of the tweet.

3) **Validation and parameter tuning: -**

The two word bags generated have different number of words. The politics.txt has 25767 words and the sports.txt has 20459 words according to the given training data, which makes politics a stronger entity than sports. On my research on this I found out that my algorithm as only errors showing sports as politics and no error on any tweet in politics category. Removing the power of politics in the code by a factor of 5 in the testing algorithm greatly increased the efficiency of code.

4) **Testing Algorithm: -**

a) Read tweet_id, and tweet from the file.

b) Divide according to category keeping tweet in a separate variable.

c) Read the word bags politics.txt and sports.txt and put their contents in a array with each line (word) as new element of the 2 separate arrays.

d) Make the elements of 2 sets unique (remove repetition)

e) If the two word sets are A and B perform the following operations on them

A=A-B

B=B-A

This removes the words that are in both the bags and have equal effect.

f) For each word in a given tweet to be categorized, all the words in both the bags are matched. If a match is found the 1 is added to the counter of that label. Moreover, if the word is a hash tag then 100 is added instead of one (giving the hash tag more value).

g) The summed factor generated for politics is reduced by 5 as discussed above in validation and parameter tuning.

h) If sports factor is more than politics factor, tweet

is marked as sports, otherwise marked as politics.

Explanation of results on validation data:

I achieved 83.5272 accuracy on the validation data and my algorithm is not based on any standard ML algorithms or techniques.

Why do you think your algorithm got the accuracy that it did on the validation data? Is scope for improvements?

My algorithm got this accuracy because I was able to realize that the training data has more words for politics than sports (which can easily be noted programmatically), which makes the strength of politics more than sports. So, My algorithm would provide wrong output only for some of the sports tweets and no politics tweets. So, I reduced the politics factor generated by algorithm by 5. That reduced the number of mistakes by 15% although now I can see errors in both politics and sports but in much reduced number.

The scope of improvement lies in finding this value to be reduced from one of the bags namely politics or sports using Neural Networks.

Note: The instructions to run the program are given in instructions.txt inside the zip file.