CS 6375.002 - Machine Learning - Spring ‘17

Assignment 2 - March 26, 2017

Problem:

To implement and evaluate Naive Bayes and Logistic Regression algorithms for text classification, for the given data set (ham and spam directories) and analyze the impact of stopwords.

Implementation Files:

No.	File Name	Usage
1	Main.java	Main Program to run
2	TrainingSet.java	Read and store Training Data and Stopwords
3	NB.java	multinomial Naive Bayes algorithm for text classification
4	LR.java	MCAP Logistic Regression algorithm with L2 regularization
5	Email.java	Read and pre-process Email
6	LexCount.java	Store lexicons with count

(The steps to execute the code are given in ReadMe.txt)

Naïve Bayes:

The multinomial Naive Bayes algorithm for text classification has been implemented based on the reference from http://nlp.stanford.edu/IR-book/pdf/13bayes.pdf (Figure 13.2). All the calculations are done in log-scale to avoid underflow. The algorithm learns from the training set and reports the accuracy on the test set (i)with stopwords and (ii)without stopwords

Equations:

- ➢ $P(Y=ham/X_1= x_1, X_2=x_2,\ldots. X_n=x_n) = P(X=x_1/Y= ham) * P(X=x_2/Y= ham) *\ldots\ldots. P(X=x_n/Y= ham) *P(Y= ham)$

- ➢ $P(Y=spam/X_1= x_1, X_2=x_2,\ldots. X_n=x_n) = P(X=x_1/Y= spam) * P(X=x_2/Y= spam) *\ldots\ldots. P(X=x_n/Y= spam) *P(Y= spam)$

Accuracy using Naive Bayes	
With Stopwords	**Without Stopwords**
72.803	72.8033472

Observations:

- Initially, the accuracy was very low without any preprocessing.
  With stop words: ~16.34 Without stop words:~ 47.68
- It gradually increased when unnecessary punctuations were removed.
  With stop words: ~54.58 Without stop words: ~62.29
- The accuracy was still better (~72-73 as in the above table) when the features were case-insensitive.

Overall, the accuracy is better for Naive Bayes without stop words.

Reason: Ham has more files than Spam. Because of the stop words, there is a possibility that it might classify all documents to Ham. Naive Bayes takes total occurrences of words into account.

Logistic Regression Implementation:

The MCAP Logistic Regression algorithm with L2 regularization has been implemented. The algorithm learns from the training set and reports accuracy on the test set for different values of λ(the regularization parameter). Gradient ascent is used for learning the weights. The gradient ascent is not run until convergence.

Computation:

$h(x) = 1/(1+\exp(-\Sigma(w_i x_i)))$, i : 1 to n
$P(Y=Ham/X) = 1/1+\exp-(w_0+ \Sigma(w_i x_i))$, i : 1 to n
$P(Y=Spam/X) = 1 – P(Y=Ham/X)$

w_i - weights assigned to the feature x_i
x_i – feature (word)

```
If h(x)>=0.5, then
    sample belongs to ham
else
    sample belongs to spam
```

The weights (w_i) of the features are:

$$w_0 = w_0 – (a/m)\ \Sigma(h(x)-y)x_0$$

$$w_i = w_i – (a/m)\ (\Sigma((h(x)-y)x_0) + l *w_i)$$

(where; a - Learning rate and l – regularization factor)

Case No:	No: of Iterations	Learning Rate	Regularization Factor(λ)	Accuracy	
				with stopwords	**without stopwords**
1.	5	0.07	0.06	74.47698744769	78.4518828451882
2.	10	0.003	0.04	73.64016736401	73.0125523012552
3.	15	0.1	0.2	76.98744769874	82.4267782426778

**Note: In the code, the number of iterations has been set to 5 by default for which Logistic Regression computation takes approximately 6 mins. Please wait for the output to be displayed.

From the above table, we see that for Logistic Regression, the accuracy is better without stopwords.

Conclusion:

The Naive Bayes and Logistic Regression algorithms have been improved by throwing away (i.e., filtering out) stop words such as \the" \of" and \for" from all the documents. The accuracy for both Naïve Bayes and Logistic Regression for this filtered set has been reported. We see an improvement in accuracy if stopwords are removed. The observations and explanations have been provided.