Implementing Naive Bayes Classifier using Spark MapReduce

Priors for each class:

	A^B_C sentiment	1.2 prior	1.2 log_prior
1	positive	0.5023891993022022	-0.6883801622634533
2	negative	0.4976108006977979	-0.6979370322103389

Evaluation:

Accuracy	*0.66*
Precision	*0.59*
Recall	*0.68*
F1 measure:	*0.64*

Algorithm description:

---

function TRAIN NAIVE BAYES(D, C) **returns** log $P(c)$ and log $P(w|c)$

for each class $c \in C$ # Calculate $P(c)$ terms
   N_{doc} = number of documents in D
   N_c = number of documents from D in class c
   $logprior[c] \leftarrow \log \dfrac{N_c}{N_{doc}}$
   $V \leftarrow$ vocabulary of D
   $bigdoc[c] \leftarrow$ **append**(d) **for** d $\in$ D **with** class c
   for each word w in V   # Calculate $P(w|c)$ terms
     $count(w,c) \leftarrow$ # of occurrences of w in $bigdoc[c]$
     $loglikelihood[w,c] \leftarrow \log \dfrac{count(w,c) + 1}{\sum_{w' \ in \ V} (count \ (w',c) + 1)}$
return $logprior$, $loglikelihood$, V

function TEST NAIVE BAYES(*testdoc, logprior, loglikelihood*, C, V) **returns** best c

for each class $c \in C$
   $sum[c] \leftarrow logprior[c]$
   for each position i in *testdoc*
     $word \leftarrow testdoc[i]$
     if *word* $\in$ V
       $sum[c] \leftarrow sum[c] + loglikelihood[word,c]$
return $\text{argmax}_c \ sum[c]$

Figure 4.2   The naive Bayes algorithm, using add-1 smoothing. To use add-α smoothing instead, change the $+1$ to $+\alpha$ for loglikelihood counts in training.