MuntinomialNaiveBayes

Algorithm:

TRAINMULTINOMIALNB($\mathbb{C}, \mathbb{D}$)
1 $V \leftarrow$ EXTRACTVOCABULARY($\mathbb{D}$)
2 $N \leftarrow$ COUNTDOCS($\mathbb{D}$)
3 **for each** $c \in \mathbb{C}$
4 **do** $N_c \leftarrow$ COUNTDOCSINCLASS($\mathbb{D}, c$)
5 $prior[c] \leftarrow N_c / N$
6 $text_c \leftarrow$ CONCATENATETEXTOFALLDOCSINCLASS($\mathbb{D}, c$)
7 **for each** $t \in V$
8 **do** $T_{ct} \leftarrow$ COUNTTOKENSOFTERM($text_c, t$)
9 **for each** $t \in V$
10 **do** $condprob[t][c] \leftarrow \frac{T_{ct}+1}{\sum_{t'}(T_{ct'}+1)}$
11 **return** $V, prior, condprob$

APPLYMULTINOMIALNB($\mathbb{C}, V, prior, condprob, d$)
1 $W \leftarrow$ EXTRACTTOKENSFROMDOC(V, d)
2 **for each** $c \in \mathbb{C}$
3 **do** $score[c] \leftarrow \log prior[c]$
4 **for each** $t \in W$
5 **do** $score[c] \mathrel{+}= \log condprob[t][c]$
6 **return** $\arg\max_{c \in \mathbb{C}} score[c]$

IMPLEMENTATION:

NaiveBayesExample.java: The main class is in this file. Initially for each file in the training spam and ham folders, the class is assigned as spam and ham respectively.

NaiveBayes.java:This predicts the class of the file

NaiveBayesKnowledgeBase.java:the knowledge gained from training the data is saved in the form of maps.

Document.java:it is used to store the data of maps as tokens.

TextTokenizer.java:`Preprocess the text by removing punctuation, duplicate spaces and lowercasing it.`

OUTPUT:

The program classisfies 280 out of the 348 files in ham as ham.

It classifies 104 out of 130 files of spam as spam.

MCAP LOGISTIC REGRESSION WITH L2 REGULARIZATION

IMPLEMENTATION:

Train.java: this file has the main class.

The L2R_LrFunction.java, L2R_L2_SvrFunction.java and L2R_L2_SvcFunction.java are used to implement L2 regularization.

The Predictor.java file is used to predict the class of the file.

OUTPUT:

The program classifies 292 out of 348 files of ham folder as ham.

It classifies 102 out of 130 files of the spam folder as spam.