Naive Bayes Classifier

Naive bayes is a parametric supervised learning technique. It is also Generative classifier which calculates prior (probability of the class itself) and the likelihood for the feature in the class and selects the class that maximizes the probability of input features. In this case we consider feature set as each word that is occurring in the document and we compute the joint probability by adding log values of all. Once done, we choose the class that maximizes the probability of the document.

Assumptions made

1. Order of words does not matter
2. Position of a word in the document does not matter
3. Features are independent given class - In this case it means that one word's occurance is independent of the other.
4. Word frequency is the feature used for classification

Decision rule:

$$y^* = \arg \text{Max}_y \; P(y) \; \Pi i \; P(X_i \mid y)$$

In case of multinomial Naive bayes regression, we add logs of conditional probabilities of each word and add the probability of the given class and choose the class that maximizes the probability.

Analysis:

Naive bayes Maximum Likelihood training is done by done by evaluating closed form expression which is done in linear time and hence executes much faster than other iterative training classifiers like Logistic regression.

In order to address the problem of "unseen words" we used the laplace smoothing and calculated probabilities of unknown words. There are alternate smoothing approaches that could be used but since data is not exactly sparse, laplace - add one smoothing works well here.

We can clearly see that use of stop words gives us much better Test accuracy and help a little reduce overfitting on training data.

How to execute Naive Bayes:

Execute the naiveBayes Jar. Provide it path to training folder, test set folder and stopWords file. It executes the Naive Bayes classifier and prints the accuracy found on the set. It also executes the trained model on training set again and confirms the accuracy on training set in order help us get insight into whether it is overfitting the data or not.

Please find detailed steps for executing the naive bayes jar below.

Steps :

1. Extract the zip file
2. Goto the path where naiveBayes.jar is located in the folder
3. Execute the command "java -jar naiveBayes.jar"
4. Provide it path to the "train" folder which has ham/spam folders for classification.
5. Provide path to test folder with same folder structure as that of train
6. Provide path to stopWords file
7. As you can see in following execution log, it prints accuracy on all data sets it comes across.

Execution Log

```
S:\ML\Source_code\Machine_learning\NaiveBayesTextClassification>java -jar naiveBayes.jar
Please enter path to training folder:
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train
Training on
 S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\ham
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\spam
Prediction on Training data:
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\ham
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\spam
Total accuracy found : 99.35205183585313
Please provide path to Test folder on which prediction needs to be made:
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\test
Prediction on Test data:
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\test\ham
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\test\spam
Total accuracy found : 94.76987447698745
Please provide path to stopWords file:
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\stopwords.t
xt
Stopwords File found: Loaded 499 stop words in memory
Training on
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\ham
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\spam
Prediction on Training data:
```

```
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\ham
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\train\spam
Total accuracy found : 99.13606911447084
Prediction on Test data:
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\test\ham
S:\ml\Source_code\Machine_learning\NaiveBayesTextClassification\src\main\resources\test\spam
Total accuracy found : 96.02510460251047
```