**Text Categorization System**

**Overview**

Text Categorization is the process of assigning predefined categories to the free-text documents. This has huge number of applications in the real world. Some of these include news stories categorization based on topics, Health report classification based on disease analysis, taxonomies of disease categorization, email spam filtering etc.

In this project, Naïve Bayes Classification model is built based on the 20 newsgroup dataset. Then the developed model is rigorously tested and improved based on various methods such as stemming, removal of stopwords etc.

This model can be used to test any text classification. Any new document will be classified into any of these 20 categories. Model can be trained on any type of Dataset.

**Data Description:**

There are 20 newsgroup data set each consisting of around roughly 700 files. TheData set is obtained from:  <http://www.hlt.utdallas.edu/~yangl/cs6375/project/>. The details about each class and the number of documents are as follows.

|  |  |  |  |
| --- | --- | --- | --- |
| ClassName | |  | | --- | | Number of Files | | Class Number |
| |  | | --- | |  | | alt.atheism | | comp.graphics | | comp.os.ms-windows.misc | | comp.sys.ibm.pc.hardware | | comp.sys.mac.hardware | | comp.windows.x | | misc.forsale | | rec.autos | | rec.motorcycles | | rec.sport.baseball | | rec.sport.hockey | | sci.crypt | | sci.electronics | | sci.med | | sci.space | | soc.religion.christian | | talk.politics.guns | | talk.politics.mideast | | talk.politics.misc | | talk.religion.misc | | |  | | --- | |  | | 480 | | 584 | | 591 | | 590 | | 578 | | 593 | | 585 | | 594 | | 598 | | 597 | | 600 | | 595 | | 591 | | 594 | | 593 | | 599 | | 546 | | 564 | | 465 | | 377 | | |  | | --- | | 0 | | 1 | | 2 | | 3 | | 4 | | 5 | | 6 | | 7 | | 8 | | 9 | | 10 | | 11 | | 12 | | 13 | | 14 | | 15 | | 16 | | 17 | | 18 | | 19 | |

**Methodology:**

General solution in Bayes estimate(smoothing) is :

+mp/ n+ m

n- number of training examples for which v = vj

nc= number of examples for which v = vj and a = ai

p = prior estimate for P(ai|vj)

m = weight given to prior.

The Naïve Bayes Conditinal Independence is assumed as follows

P(doc|vj) =

To simplify,

**(w|c) = (count(w,c) + 1) /( count(c) + |v|)**

Where count(w,c) means the number of times the word has occurred in class C

1: smoothing factor

Count(c) = total number of words in class C

|v| = total number of distinct words in the entire classification.

After calculating the probability of each word in the test document the probability for the entire document is multiplied by the class probability

= Nc/N

Nc = Number of documents with class C

N = total number of documents.

**Data Structures and Algorithms**

HashMaps are used for storing most of the data required. First, each documents are given unique document IDs and the word Map consisting of each word and the frequency.

Thus we have HashMap<<docId, HashMap<word, frequency>> for each word.

Then the classDocument Map is created which represents the document ID’s for each class. i.e HashMap<class, List<documents>> ex: class 1 <doc13,doc 25,doc34…>

After calculating the word probability for each class, the result is stored for its probability in each class using HashMap again. This would look like

HashMap<word HashMap<classs, probability>

Example: <science <class 2, 0.234>> etc..

Major algorithmic steps used to build the training Model:

1)Read all the document and build vocabulary consisting of words and its frequency for each document.

2) Remove stop words and stem each word before building vocabulary

3)Build classDocument map consisting of document Ids for each class. This helps in knowing number of distinct words, total words for each class.

3) For each word encountered in each document, Calculate the probability of belonging to each class. Store the result appropriately in the hashMap

4)Serialize the necessary hashMaps such as vocabulary, probabilities etc so that training model can be built directly from Index files in the subsequent testings.

Major steps that are used to classify the document:

1. Remove all the non-alphanumeric characters and convert all the characters to lowercase.
2. Remove the stop words for improving performance
3. Stem each word before storing it to improve the performance.
4. Calculate the probability of each word belonging to each class and store the result appropriately.
5. Log likelihood is calculated and the document’s probability for each class is evaluated.
6. Document is classified based on the maximum likelihood.

**Experiment:**

The experiment is carried out in different stages of training and testing phases.

During training the model, different strategies are used to come up with best training model.

* Removal of stopwords

Removal of stopwords showed an accuracy of around 76.3% when run on the entire dataset using the 2000 text files provided.

However, The observation was made that without removing stopwords the classification would be inclined towards the class having more stopwords with similar other keywords of neighbouring class.

For example, the document belonging to class rec.sports.hockey would be misclassified as rec.sports.baseball since second class has more stopwords compared to former class. Removing stop-words would definitely avoid such mis-classifications.

**Stemming:**

Stemming has shown a certain improvement in the over all classification accuracy. In order to implement stemming, an open source stemming class has been taken and modified according to the need.

The Stemmer.java is taken and modified from open-source

[http://chianti.ucsd.edu/svn/csplugins/trunk/soc/layla/WordCloudPlugin/tags/v1.00\_RC2/src/cytoscape/csplugins/wordcloud/Stemmer.java](http://chianti.ucsd.edu/svn/csplugins/trunk/soc/layla/WordCloudPlugin/tags/v1.00_RC2/src/cytoscape/csplugins/wordcloud/Stemmer.java%20)

The following table shows the effect of removing stop-words and stemming

|  |  |  |
| --- | --- | --- |
| ClassName | Before removing stop-words and stemming  Correctly classified Total number of files | |
| alt.atheism | 74 | 100 |
| comp.graphics | 84 | 100 |
| comp.os.ms-windows.misc | 0 | 100 |
| comp.sys.ibm.pc.hardware | 81 | 100 |
| comp.sys.mac.hardware | 74 | 100 |
| comp.windows.x | 84 | 100 |
| misc.forsale | 64 | 100 |
| rec.autos | 90 | 100 |
| rec.motorcycles | 96 | 100 |
| rec.sport.baseball | 88 | 100 |
| rec.sport.hockey | 93 | 100 |
| sci.crypt | 100 | 100 |
| sci.electronics | 95 | 100 |
| sci.med | 81 | 100 |
| sci.space | 94 | 100 |
| soc.religion.christian | 92 | 100 |
| talk.politics.guns | 90 | 100 |
| talk.politics.mideast | 91 | 100 |
| talk.politics.misc | 60 | 100 |
| talk.religion.misc | 35 | 100 |

**Result:**

After removing stopwords and stemming ,the result for the entire test data increased from 78.3 to 79.45%.

Also its observed that stemming and removing stopwords had no affect for certain documents. These documents are mostly the documents containing very few stopwords or The document has the strong keywords specific to the class it belongs to.

The following table shows the final result of accuracy of 79.45% for the entire test set consisting of 2000 documents.

|  |  |  |
| --- | --- | --- |
| ClassName | After removing stop-words and stemming  Correctly classified Total number of files | |
| alt.atheism | 76 | 100 |
| comp.graphics | 86 | 100 |
| comp.os.ms-windows.misc | 6 | 100 |
| comp.sys.ibm.pc.hardware | 83 | 100 |
| comp.sys.mac.hardware | 78 | 100 |
| comp.windows.x | 84 | 100 |
| misc.forsale | 69 | 100 |
| rec.autos | 94 | 100 |
| rec.motorcycles | 96 | 100 |
| rec.sport.baseball | 88 | 100 |
| rec.sport.hockey | 99 | 100 |
| sci.crypt | 100 | 100 |
| sci.electronics | 97 | 100 |
| sci.med | 89 | 100 |
| sci.space | 95 | 100 |
| soc.religion.christian | 95 | 100 |
| talk.politics.guns | 94 | 100 |
| talk.politics.mideast | 96 | 100 |
| talk.politics.misc | 62 | 100 |
| talk.religion.misc | 35 | 100 |

**Conclusion:**

In this study it has been shown that how a simplest machine algorithm like Naïve bayes can be used to achieve very good classification accuracy, which makes us wondering about the complex algorithms like SVMs, Tree based and neural network performance in this domain. Also the experimented conducted has shown that Naïve-Bayes accuracy gives very good accuracy in all the cases unlike other machine learning algorithms.

Further improvements are done by removing the stop-words from the documents and stemming all the words which has resulted in the increase of accuracy.

The Final accuracy of 79.45% has been observed during testing phase.

Thus one should consider stemming and removing stopwords while using Naïve Bays classification as it improves the accuracy with only very little constraints on the execution time.

**Future Enhancements**

Other classification methods such as K-nearest classification, Decision trees, SVMs etc can be used on the same test Data and the voting or average of these classification can be used to classify the document like in Bagging.

Also the stop-words can be defined in a more Training specific way rather than the list of general stop words.