Text Categorization

Building a kNN classifier for Reuters-21578 collection

Anita Krishnakumar
anita@soe.ucsc.edu
Introduction
Scope of the project

Phase I
• Study of different techniques for preprocessing datasets
• Study of different methods for Text Categorization
• Survey of previous works on the Reuters collection
• Presenting summary of results

Phase 2
• Preprocessing Reuters-21578 collection
• Implementation of the kNN algorithm
• Comparison of results with those in the literature
Text Categorization

The need:

• With the rapid growth of online information, there is a growing need for tools that help in finding, filtering and managing the high-dimensional data.

• Automated text categorization is a supervised learning task, defined as assigning category labels to new documents based on likelihood suggested by a training set of labeled documents.

• Real-world applications of text categorization often require a system to deal with tens of thousands of categories defined over a large taxonomy. Since building these text classifiers by hand is time consuming and costly, automated text categorization has gained importance over the years.
Phases involved in Text Categorization

- Collecting the dataset
- Preprocessing
- Indexing
- Dimensionality reduction
- Implementing the classifier
- Performance measures
Dataset Preprocessing

• The reuters-21578 collection is publicly available at http://www.daviddlewis.com/resources/testcollections/reuters21578/

• Preprocessing
  1. The documents in the data set is converted into a representation suitable for the learning algorithm. It involves the following steps:
     – Removal of HTML and other tags
     – Removal of stop-words
     – Word stemming
2. Indexing Techniques
   – Boolean weighting
   – Word frequency weighting
   – Tf x idf weighting
   – Tfc weighting
   – Entropy weighting
Preprocessing contd.

3. Dimensionality reduction
   - Document Frequency thresholding
   - Information gain
   - X2 statistics

4. Re-parameterization
   - Latent Semantic Indexing
Learning Methods for Text Categorization

- Naïve bayes
- K-nearest neighbors
- Decision trees
- Rocchio’s algorithm
- Support vector machines
- Neural networks
- Linear least squares fit
Survey of previous works on Reuters-21578 collection

<table>
<thead>
<tr>
<th>Author</th>
<th>Train</th>
<th>Test</th>
<th>Topics</th>
<th>Indexing</th>
<th>Reduction</th>
<th>Method</th>
<th>Measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joachims</td>
<td>9603</td>
<td>3299</td>
<td>90</td>
<td>tfc</td>
<td>IG</td>
<td>binary</td>
<td>break-even</td>
</tr>
<tr>
<td>Duamis</td>
<td>9603</td>
<td>3299</td>
<td>118</td>
<td>boolean</td>
<td>MI</td>
<td>binary</td>
<td>break-even</td>
</tr>
<tr>
<td>Shapire</td>
<td>9603</td>
<td>3299</td>
<td>*</td>
<td>tf x idf</td>
<td>None</td>
<td>multiclass</td>
<td>precision</td>
</tr>
<tr>
<td>Weiss</td>
<td>9603</td>
<td>3299</td>
<td>95</td>
<td>frequency</td>
<td>?</td>
<td>binary</td>
<td>break-even</td>
</tr>
<tr>
<td>yang</td>
<td>7789</td>
<td>3309</td>
<td>93</td>
<td>ltc</td>
<td>$\chi^2$</td>
<td>binary</td>
<td>break-even</td>
</tr>
</tbody>
</table>

Table 1: Summary of previous work.

<table>
<thead>
<tr>
<th>Author</th>
<th>Rocchio</th>
<th>Bayes</th>
<th>kNN</th>
<th>Tree</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>Joachims</td>
<td>79.9</td>
<td>72.0</td>
<td>82.3</td>
<td>79.4</td>
<td>86.0</td>
</tr>
<tr>
<td>Duamis</td>
<td>61.7</td>
<td>75.2</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Shapire</td>
<td>x</td>
<td>x</td>
<td>-</td>
<td>-</td>
<td>x</td>
</tr>
<tr>
<td>Weiss</td>
<td>78.7</td>
<td>73.4</td>
<td>86.3</td>
<td>78.9</td>
<td>86.3</td>
</tr>
<tr>
<td>yang</td>
<td>75.0</td>
<td>71.0</td>
<td>85.0</td>
<td>79.0</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 2: Summary of previous works(2) An x in the table signifies that the method was tested but a different performance measure other than break-even point was used.
My experiments on the Reuters-21578 collection

• **Preprocessing**
  I use Matlab to remove all the SGML tags and extract each document from the Reuters file and write it into an individual file, so that it is can be indexed. The individual words were extracted and then stop words were removed using a list of frequent English words. Word stemming was performed using the Porter stemmer. I use Lucene for stemming and to extract the word vectors from each document.

• **Indexing**
  Tf-idf weighting is used to index the documents. The weight \( a_{ik} \) for word \( i \) in document \( k \) is given by

\[
a_{ik} = f_{ik} \times \log\left(\frac{N}{n_i}\right)
\]

where \( f_{ik} \) is the frequency of the word \( i \) in document \( k \), \( N \) is the number of training documents and \( n_i \) is the total number of times word \( i \) occurs in the whole collection.
• **Dimensionality reduction**

Feature selection is performed here using Document Frequency Thresholding. Words occurring in just one document are removed based on the assumption that rare words do not affect category prediction.

• **Method used**

The kNN method is a very simple approach and from the survey of various previous works, we see that kNN showed good performance on text categorization tasks. Hence, this method was chosen for classification. Through investigations on suitable choices for $k$ we see that the performance of kNN is relatively stable for a large range of $k$ values. The value of $k$ chosen for experiment here is 25.
The kNN Algorithm

- KNN using average cosine
  - Select k nearest training documents, where the similarity is measured by the cosine between a given testing document and a training document.
  - Using cosine values of k nearest neighbors and frequency of documents of each class i in k nearest neighbors, compute average cosine value for each class i, \( \text{Avg}_\text{Cosine}(i) \).
  - Assign (i.e., classify) the testing document a class label which has largest average cosine.
Experimental results

The K-nearest neighbour algorithm was used for classification. It is a very simple approach, but has shown to have approximately the same performance as more complicated methods. The results gave a precision/recall breakeven point of approximately 79.3\% which is comparable to the other studies reported for the Reuters-21578 collection.

<table>
<thead>
<tr>
<th>Topic</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>earnings</td>
<td>0.94</td>
<td>0.91</td>
</tr>
<tr>
<td>acquisitions</td>
<td>0.93</td>
<td>0.89</td>
</tr>
<tr>
<td>money-fx</td>
<td>0.87</td>
<td>0.60</td>
</tr>
<tr>
<td>crude</td>
<td>0.88</td>
<td>0.70</td>
</tr>
<tr>
<td>grain</td>
<td>0.81</td>
<td>0.78</td>
</tr>
<tr>
<td>trade</td>
<td>0.89</td>
<td>0.66</td>
</tr>
<tr>
<td>interest</td>
<td>0.80</td>
<td>0.71</td>
</tr>
<tr>
<td>ship</td>
<td>0.85</td>
<td>0.77</td>
</tr>
<tr>
<td>wheat</td>
<td>0.67</td>
<td>0.52</td>
</tr>
<tr>
<td>corn</td>
<td>0.31</td>
<td>0.63</td>
</tr>
</tbody>
</table>

Table 3: Summary of recall and precision for the 10 most frequent topics.