Keywords Extraction, Document Similarity and Categorization

Huaizhong KOU and Georges Gardarin

PRiSM Lab, University of Versailles

45 Avenue Etats-Unis ,78035 Versailles, France

{Firstname.Lastname@prism.uvsq.fr}

Abstract: With the advent of Internet since 1990's, we have seen a tremendous growth in the volume of online text documents available on the Internet, news sources, and company-wide intranets. There is increasingly need for tools to deal with text documents. This white paper briefly presents the different perspectives of document categorization, the document similarity, and keywords extraction also is approached. We discuss the popular algorithms.

1 Introduction

With the advent of Internet since 1990's, we have seen a tremendous growth in the volume of online text documents available on the Internet, news sources, and company-wide intranets. In fact, unstructured text data such as electronic emails and Web pages become gradually the predominant data type. This provides a huge opportunity to make more effective use of these documents, so there is increasingly need for tools to deal with text documents. To meet such increasingly needs, some product for analyzing text documents can be developed. All techniques involved in document analysis have formed a new exciting research area often called as Text Mining. Text Mining, which is more challenging than traditional Data Mining, aims to facilitate user to understand text documents, to organize the documents, to analyze and compare the documents, etc. It will combine the techniques from many disciplines, for example, data mining, natural language processing, and artificial intelligent, machine learning. Among its different functions addressed by researchers are information retrieval, information extraction, document categorization, document summarization, etc..

The white paper will only address document categorization, document similarity, and keywords extraction. Document categorization focus on trying to assign a document to one or multiple predefined categories while document similarity aids users to compare the documents based on their contents. Since a keyword is often a phrase of two or more words, some researchers prefer to call keywords as key phrases. In this paper, we use keywords and key phrases alternatively. Keywords extraction, which may be viewed as a classification problem, is an important component of the document processing application system.

2 Index Language and Document Vector Model

Document representation is a common basic problem encountered by Text Mining community. A full free natural language text can not be directly proceeded by text mining methods. Given a document of specific domain, it must be represented in a form suitable for existing text mining methods. It involves two problems: One, selecting an optimal set of important terms from domain-specific document corpus; Second, representing a document by using selected terms. The former aims to generate an Index language and the later to represent documents by index language.

This section approaches index language automatic generation and document vector representation model. Using automatic methods, a domain-specific index language can be automatically constructed from a document corpus while every document can be represented as a vector whose elements measure importance of index language terms in the document. After documents are represented as vectors, many text mining methods can perform. For example, document clustering, document similarity analysis, and document categorization, etc..

2.1 Index Language Generation

2.1.1 Tokenizing document

Documents are considered strings. They contain non-alphabet characters that are language-specific. All non-alphabet characters must be removed while the others are kept and are converted into low case (case folding). Then a document string is tokenized by using white space as delimiter. A tokenized document only contains language-specific alphabets in low case.

2.1.2 Stop-words

Stop-words, which are language-specific functional words, neither characterize document contents nor discriminate the documents. They must be removed. Removal of stop-words can expand words and enhance the discrimination degree between documents and can improve the system performance.

2.1.3 Stemming word*

Stemming converts words to their stems, which incorporates a great deal of language-dependent linguistic knowledge. Behind stemming, the hypothesis is that words with the same stem or word root mostly describe same or relatively close concepts in text and so words can be conflated by using stems. Word stemming can reduce the number of document terms and expand terms. So it increases the system recall while the precision is decreased to some extent. The Porter (1980) and Lovins (1968) stemming algorithm are two commonly used algorithms. They both use language-dependent heuristic rules to transform word suffixes. An alternative to heuristic rules is to use a dictionary that explicitly lists the stem for every word. In the practice, stemming is facultative. In addition, stemming word is not simple exercise.

The document preprocessing does not consider the syntactic and semantic association of terms in documents and the document structures information about sentence and paragraph structure.

2.1.4 Item weighting

The different terms plays different rules in identifying the document contents and in discriminating one document form another. For example, the terms in document title and abstract often are more important than terms only occurring in document corpus, and the terms occurring frequently in document more significant than ones occurring only one or two times. Item weighting will assign a value to each term to indicate such importance. Reminding both the content identification and discrimination of documents, we should not only consider the number of times a term occurs but also take into account for the number of documents in which the term occurs. For example, "network" occurs 100 times in all documents while "neuron" only occurs 20 times in first document and rarely occurs in others documents. Which term is more important for first document? In practice, there exist more than 6 weighting algorithms among which most use both the term frequency and the document frequency of terms. The mostly used two term weighting algorithms are *if-idf* model and the entropy-weighting model. The entropy of terms represents the degree to which the terms bear information content of document. See **Annex A** for details.

2.1.5 Feature term selection

The feature term selection is very crucial to the document representations. The goal of feature selection is to enhance the performance of system, to minimize the use of raw feature and to reduce the dimensions of document representation space. Given a collection of training documents, all unique terms found in the collection obtained after stop-words removal and stemming words are too large to be applied directly to learning algorithm. In addition, the experimental results have shown that using all terms could not produce the satisfactory performance. So the irrelevant and redundant terms must be removed and an optimal feature term subset must be selected. The algorithms about feature selection have been investigated extensively. Sometimes, the relation between feature terms needs be analyzed and is used to reduce the dimension of feature term space. This analysis can be done by domain-specific expert or automatic technique for example term clustering.

In machine learning community, there exist two types of feature selection algorithm: wrapper and filter ([Miguel et al 99][Daphne et al 96]). The wrapper algorithm considers the performance of a particular learning algorithm while an optimal subset of feature term is selected. The result is dependant of the algorithm. The filter algorithm measures the weight of every term and uses rank criterion principle to select feature term while any machine-learning algorithm is not considered. It is dependent of the collection of documents and is independent of any algorithm.

In the text mining context, most of feature term selection algorithms are of filter. The commonly used feature selection algorithms are the DF algorithm, the IG algorithm, the CHI algorithms ([Yang et al97]) and the LSI ([Scott 90]). The DF algorithm calculates term document frequency, than determines one high threshold and one low threshold. Both terms with document frequency higher than the high threshold and ones with documents frequency lower than the low threshold are removed. The IG estimates the capability of category prediction for each term present in training documents by calculating the number of information bits of terms in predefine documents categories. If the IG value of a term is less than a prefixed threshold, then it will be removed. The CHI algorithm uses X^2 statistic method to estimate the membership between the terms and the predefine document categories. The terms with the membership less than a prefixed threshold will be removed. Different from IG and CHI, the LSI algorithm is an unsupervised algorithm and does not consider the dependency between the feature terms and the categories while it uses the singular-value matrix decomposition algorithm. LSI performs the linear transforms on the original feature terms and obtains new feature terms, which are the linear combinations of the original feature terms. Then the first *k* principal feature terms are selected. LSI can exploit the latent semantic relations between terms. Some experiments state that LSI can improve the system performance. Note that both IG and CHI algorithms are preferred for document categorization. There is not a rule to determine which algorithm is better than others. The performance of a given selection algorithm depends also on the properties of example documents.

All feature terms selected by using selection algorithms will constitute a domain-specific index language.

2.1.5.1 Thesaurus and term phrases

With normal feature term selection, both high frequency and low frequency terms are mostly removed. This will induce two problems: first, removal of the high frequency terms may reduce the recall of a system; second, removal of the low frequency terms perhaps reduces the precision of a system. Instead of simply discarding such terms, the associations between them can be exploited to supplement to the selection algorithm with the hope of refining or broadening the interpretation of them. The resulting associations furthermore are incorporated into document indexing. A domain-specific thesaurus and term phase based on co-occurrence analysis of terms may ease these two problems.

A thesaurus provides a grouping of the terms used in a given topic area into thesaurus classes to which class identifiers are assigned. The thesaurus class identifiers can replace the original terms in document, so it broadens terms and enhances the recall. Granted a document collection, the domain-specific thesaurus can be constructed automatically for example by clustering terms based on term by document matrix ([Salton83]). The term phrases are combinations of two or more terms with high co-occurrence frequency. Various phrase-generation methods are possible, including the use of term co-occurrence statistic analysis in documents.

Now, the low frequency terms can be used to index document by replacing them with their thesaurus class identifiers if relative thesaurus class identifiers exits; otherwise they can be discarded. As for the high frequency terms, replacing them with term phrases if relative terms phrases are generated can use them; otherwise they can also be discarded.

So term phrases can be used to supplement word-based indexes. However, experimental evidence for their effect on precision and recall is mixed ([Zhai et al 96]).

2.2 Document vector model

The document vector representation model is widely accepted in the community. By the vector model, a document is mapped into a point in the *T* dimension Euclidean space where each feature term of index language constitutes one axis and its coordinate value is the weighting value of corresponding feature item in the document. Formerly, given a document, it can be represented by

$$d_j=(w_{1j},w_{2j},w_{3j},\ldots w_{Tj}) \in R^T$$

Where w_{ij} is the weight of the ith term in jth document d_j, $1\le i\le T$ and $1\le j\le N$. Term weight can be calculated by using methods described in **Section 2.1.4**. In order to eliminate the difference of document length, the document vector usually furthermore is normalized to unit length.

We observe that the term order and term structure of document are lost in the vector model and that the document is only taken as a bag of term-weight pairs.

3 Document Similarity

The notion of document similarity between documents is crucial. Because a document can address multiple area topics, the understanding of the similarity between documents is closely related to the discussing problems. In other word, it is domain-dependent. For example, given two documents, the first concerns both bay fight and oil while the second only involves oil. If one only is concerned about the fight problem, the similarity between these two documents is very weak. If one discusses the problem of word economy and old, they are similar.

A very simple similarity measure would be the degree of overlap for single words in the documents. However, due to the ambiguity of single words, this measure of similarity is rather imprecise.

One alternative to using single words would be to perform a semantic analysis of the documents, in order to find the concepts mentioned in the text and use them to calculate the document similarity. This kind of analysis is very expensive and furthermore depends on a lot of domain-dependent knowledge that has to be constructed manually or obtained from other sources.

Another approach to this problem is to use lexical affinities between words. A lexical affinity is a correlated group of words, which appear frequently within a short distance of one another. Lexical affinities include phrases like “online library” or “computer hardware” as well as other less readable words groupings. They are generated dynamically, thus they are specific for each collection. A set of semantically rich terms can be obtained without a need to hand-code a specialized lexicon or a thesaurus ([IBM98]).

In practice, given two documents, we will first transform them to two document vectors for example d_1 and d_2 by using the methods introduce in **Section 2** as following:

$$d_1=(w_{11},w_{21},w_{31},\ldots w_{T1})$$

and

$$d_2=(w_{12},w_{22},w_{32},\ldots w_{T2})$$

Then similarity between these documents is measured by the cosine function value of the two vectors:

$$Similarity(d_1,d_2)=\cos(d_1,d_2)=\frac{d_1 \bullet d_2}{\|d_1\|_2 \times \|d_2\|_2}$$
$$=\frac{\sum w_{1l} \times w_{2l}}{\sqrt{\sum w^2_{1l}} \times \sqrt{\sum w^2_{2l}}}$$

4 Keywords Extraction

A keyword is a single word or multiple-words present in the documents that can characterize and summarize the topics covered by documents. Keyword extraction is a procedure of selection important, topical phrases from documents. It is distinct from information extraction. The later involves extracting specific types of task-dependent information while the former is not specific and aims to produce topical phrases for any type of factual document.

4.1 Keyword for a document

There are two main approaches to automatic extraction of key phrases from text: syntactic analysis, where useful sequences are identified based on lexical patterns; and statistical analysis, where techniques such as frequency counts are used to find word pairs ([Gutin et al 98]). By statistic analysis, keywords extraction may be viewed as a classification problem, where a document can be seen as a bag of phrases, where each phase belongs to one of two possible classes: either it is a key-phrase or it is a non-key phrase ([Turney97], [Turney99] [Gutin et al 98]). Decision tree algorithm C4.5 is implemented in [Turney99] while [Gutin et al 98] used the naïve Bayes algorithm. One observation is that key-phrases often correspond to frequent noun phrase in the text ([Turney97]).

Granted a collection of example documents, of which every phrase has been manually classified as a key-phrase or non-keyphrase, the starting point of learning algorithm is to tag all noun phrases in a document by using one tagging algorithm for example Eric Brill's Tagger algorithm ([Brill92]). In [Turney99], a document is represented as a set of feature vectors by making a list of all phrases present in the document. Then using stemming converted these phrases into their stemmed forms. For each unique stemmed phrase, a feature vector is constructed. The feature vector contains 12 features that describe the position and occurrence frequency of the stemmed phrase. A decision tree of C4.5 can be created by using these feature vectors. The leaves of the tree attempt to predict class feature of feature vector. If the class value of a vector is 1, then the phrase is a keyphrase. Different from [Turney99], the naïve Bayes algorithms is used and only three attributes are calculated for each phrase: where in the document it first occurs, how frequently it is used, and how rare it is in general usage ([Gutin et al 98]). Then a keyphrase model can be built by the naïve Bayes algorithm ([Langley et al 92]). Now given a new document, the three attributes are calculated for candidate phrases found in the document, then the naïve Bayes algorithm use these values and the keyphrase model to calculate an evidence score for each phrase. The phrases are ranked by the evidence score and the top phrases can be selected as keyphrases.

Keyword extraction from a document has many potential applications: it can create a metadata for a document, it can facilitate skimming document by highlighting keywords, it can also be used as index term for searching in document collections, and it may be used to analyze usage patterns in web server logs.

4.2 Keyword for document group

5 Categorization models

Categorization is one of the key techniques to handle and organize text data. It is the procedure of assigning one or multiple predefined category labels to a free text document (category sometimes called "topics" or "themes") based on their contents. It is widely used in a variety of natural language processing applications. For example, building a personalized net news filter by learning about the news-reading preferences of user, assigning e-mail to a set of folders, transferring user's feedback information to a expert, analyzing online financial newswires, organizing users by analyzing their profiles, guiding a user's search on the World Wide Web etc.

The document categorization is a supervised machine learning system. It starts with the preparation of a collection of example documents and is achieved by learning from the collection of example documents.

In literature, the various categorization learning algorithms have been developed. Among them are similarity-based algorithm, Bayesian probabilistic algorithm, decision tree algorithm, neural networks algorithms and Support Vector Machine models. Each of them performs relatively well in certain domain-specific environments. In practice, given an application it is advantageous to be able to try different approaches to decide which kind of categorization algorithm suitable for a specific real word situation. Next we will introduce 6 algorithms.

5.1.1 Centroid-based algorithm

It calculates a centroid vector for each category by averaging all document vectors in the category. All such centroid vectors represent the learned model. The similarity between two documents d_i and d_j is defined as cosine function cosin (d_i ,d_j). Given a new document *d*, the similarities Similarity(*d*, $\vec{c}_i$) between *d* and every prototype vector $\vec{c}_i$ are calculated. And the document *d* will be put in such class that the similarity between it and this new document *d* is maximum or more than a user predetermined threshold. IBM has used this classifier in their product Intelligent Miner for Text ([IBM98]). Note that the similarity between the new document and the centroid vector represents the average of the similarity between the new document and every document in the category.

5.1.2 k-NN

k-Nearest Neighborhood, a top-performing method, is an instance-based learning method and has been intensively studied in pattern recognition for over four decades. First it computes the similarity between a new document and the training documents and selects the k top-ranking nearest documents, then calculate similarity scores for each category, finds out the category with the highest category score. There are two methods for calculating the category score: majority voting and similarity score summing ([Tuba Y. et al98]).

5.1.3 Naive Bayesian algorithm

The naïve Bayesian classifier uses a probabilistic model of text. It treats a document as the *T* time dependent Bernoulli tests, which observes the presence of document term in every category. *T* is the number of terms present in the document. The relevance between the document and the categories can be estimated by such Bernoulli tests (See **Annex B**). Then the category with the highest relevance value will be assigned to the document. This model assumes that words or terms occur in the collection documents independently each other ([Eui-hong et al 00]).

5.1.4 SVM

Support Vector Machine (SVM) is a relatively new learning approach introduced by [Vapnik95] for solving two-class pattern recognition problems. It is based on the Structural Risk Minimization principle. The SVM problem is to find the decision surface that maximizes the margin between the data points in a training set ([Yang et al 99]). See Figure1 and Figure2.

The dashed lines parallel to the solid ones show how much one can move the decision surface without causing misclassification of the data. The distance between each set of those parallel lines is referred to as " the margin". Formerly saying, let *S* be a set of points $x_i \in R^n$ with *i*=1,2,…N. Each point x_i belongs to either of two classes and thus is given a label $y_i \in \{-1,1\}$. The goal is to establish the equation of a hyper plane that divides *S* leaving all the points of the same class on the same side while maximizing

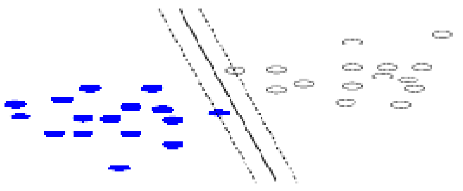
Figure1: A decision line (solid) with a smaller margin which is the distance between the two parallel dashed lines.

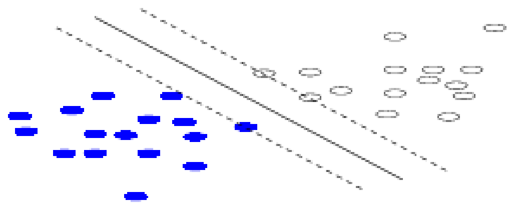
Figure2 The decision line with the maximal margin. The data points on the dashed lines are the Support Vectors.

the minimum distance between either of the two classes and the hyper plane ([M.Pontil et al 97]). Obviously, SVM is a binary classifier. For the document categorization, S is the all training document vectors, the goal is to determine a hyper plane equation for each predefined category. So the document categorization is decomposed to plural binary categorization problems.

5.1.5 Decision tree C4.5

Decision tree (DT) learning is one of the most widely used and practical methods for inductive inference from examples. It is a method for approximating discrete-valued functions that is roust to noisy data and capable of learning disjunctive expressions. DT classifies instances by sorting them down the tree from the root to some leaf nod, which provides the classification of the instance ([Mitcell98]). C4.5 is a widely used decision tree algorithm evolved by Qinlan ([Qinlan93]) that has been shown to produce good classification results primarily on low dimensional data sets.

C4.5 uses *divide-conquer* strategy by Hunt to construct a decision tree from a set D of training cases – training documents. A leaf indicates a class - category and a decision node specifies a test to be carried out on a single attribute value, which maximizes the information gain, with one branch and subtree for each possible outcome of the test.

5.1.6 Neural networks

Artificial neural networks (ANNs) provide a general, practical method for learning real-valued, discrete-valued, and vector-valued functions from examples. In rough analogy to a biological system, artificial neural networks are built out of a densely interconnected set of simple units, where each unit takes a number of real-valued inputs and produces a single real-valued output. See Figure3. ANNs learning is well suited to problems where the training data corresponds to noisy data, instances are represented by many attribute-value pairs and the target function output may be discrete-valued, real-valued or discrete-valued attributes etc.

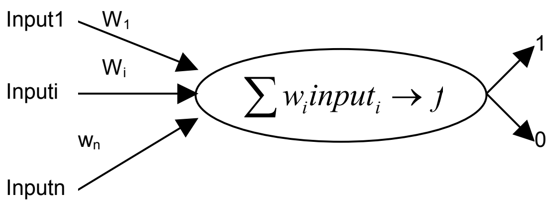

Figure3: ANN module

ANNs learning algorithms have been applied to the document retrieval community ([E.Wiener95], [Miguel et al 99], [Wilkinson et al 91]) where the documents are represented by vectors with elements as the values of document terms and the training data is a collection of example documents associated with one predefined category.

6 Conclusion

In this paper, we described document categorization that is a problem of supervised machine learning from examples. One categorization system can be built in two phrases. First, the system is learned from the training part of the example documents; second it is evaluated and tuned by using the test part of the example documents. We indicated that the term weighting models and the feature term selections are two crucial components. The 6 learning algorithms have been presented. The choice of classifiers is related to an application circumstance. No rules accepted determine which a method outperforms over another. Document similarity concept is explained. We claimed that it is domain-dependent and can be calculate by the cosine function.

Finally, we addressed the keyword extraction and shown its applications. The phrase tagging and the phrase representation are introduced. Considering it as a supervised machine learning problem, C4.5 algorithm and the naïve Bayesian algorithm are discussed for key word extraction.

Reference

[Brill92] Brill, E. (1992). A simple rule-based part of speech tagger. In Proceedings of the Third Conference on Applied Natural Language Processing, Association for Computational linguistics (ACL), Trento, Italy.

[Daphne et al 96] Daphne Koller and Mehran Sahami, Toward Optimal Feature Selection, *Proceedings of the 13th International Conference on Machine Learning (ML)*, Bari, Italy, July 1996, pp. 284-292.

[EWiener95]E.Wiener, J.O. Pedersen, and A.S.Weigend. A neural network approach to topic spotting. In Proceedings of the Fourth Annual Symposium on Document Analysis and Information Retrieval (SDAIR'95), 1995

[Eui-Hong et al 00] Eui-Hong H. and George K., Centroid-Based Document Classification: Analysis & Experimental Results, In European Conference on Principles of Data Mining and Knowledge Discovery (PKDD), 2000

[Gutin et al 98] C. Gutin, G. Paynter, I. Witten, C.Nevill-Manning, E. Frank, Improving Browsing in Digital Libraries with Keyphrase Indexes, 1998.

[IBM98] A White paper from IBM: technology Text Mining: Turning Information Into Knowledge. February 17, 1998, editor: Daniel Tkach.

[Karypis et al 00] G. Karypis and Eui-Hong Han, Concept Indexing: A fast Dimensionality Reduction Algorithm with Applications to Document Retrieval & Categorization, Technical Report TR-00-0016, University of Minnesota, 2000

[Kohavi et al 95] R. Kohavi and D. Sommerfield. Feature subset selection using the wrapper method: Overfitting and dynamic search space topology. In Proc. Of The First Int'l Conference on Knowledge Discovery and Data Mining, pp.192-197, Montreal, Quebec, 1995.

[Langleyet al 92] P. Langley, W. Iba, and K. Thompson, an Analysis of Bayesian Classifiers, Proceedings of the Tenth National Conference On Artificial Intelligence,PP223-228 (AAAI Press, 1992).

[Miguel et al 99] Miguel E. Ruiz and Padmini Srinivasan, Coming Machine Learning and Hierarchical Indexing Structures for Text Categorization, In Advances in Classification Research-Volume 10: Proceedings of the 10th ASIS SIGCR Classification Research Workshop. Washington DC, 1999.

[Mitcell98] Mitcell, T. (1998). *Machine Learning*. Boston, MA, McGraw-Hill.

[M.Pontil et al 97] M. Pontil and A. Verri, Properties of Support Vector Machines, C.B.C.L paper No.152 august, 1997

[Qinlan93] C4.5: Programs for Machine learning. Morgan Kanfmann, 1993.

[Salton83] Salton and McGill: Introduction to Modern Information retrieval, 1983, McGraw-Hill Book Company

[Scott 90] Scott D., Indexing by Latent Semantic Analysis, Journal of the American Society for Information Science, Vol. 41,No, 6,pp.391-407,1990

[Turney97] P.Turney, Extraction of Keyphreases form Text: Evaluation of Four Algorithms, the technique report: ERB-1051

[Vapnik95] V. Vapnik, The Nature of Statistical Learning Theory. Springer New York, 1995.

[Wilkinson et al 91] Ross Wilkinson and Philip Hingston, Using the cosine measure in a neural network for document retrieval. 1991 ACM 0-89791-48-1/91/0009/0202, pp.202-210

[Yang et al97] Yiming Yang and Jan O. Pederson, A Comparative Study on Feature Selection in Text Categorization, In the 14th Int. Conf. On Machine Learning, pp412-420, 1997.

[Yang et al 99] Yiming Y. and Xin Liu, A re-examination of text categorization methods, Proceedings of SIGR'99, pp. 42-49.

[Zhai et al 96] C. Zhai, X. Tong, N. Milic-Frayling, D.A. Evans, Evaluation of Syntactic Phrase Indexing—CLARIT NLP Track Report, in: NIST Special Publication 500-238: the Fifth Text Retrieval Conference (TREC-5), pp.347-358 (United States Department of Commerce, National Institute of Standards and Technology, 1996).

Annex A: Term Weighting Models

Some studies show that an appropriate term weighting improves system performance at most 40%. We present the four approaches to term weighting.

- **Word frequency weighting**

$w_{ij} = f_{ij}$

It does not take into account the frequency of term in other documents and different length of documents. Beside, it neglects document discrimination. According to it, the high is a term frequency, the important is the term.

tf-idf weighting

$w_{ij} = f_{ij} * \log(N/df_j)$.

Duo to consideration of document frequency of term, tf-idf weighting can measure the relevance and irrelevance of term in documents.

tfc-weighting

$$w_{ij} = \frac{f_{ij} * \log\left(\frac{N}{n_i}\right)}{\sqrt{\sum\left[f_{Ij} * \log\left(\frac{N}{n_j}\right)\right]^2}}$$

Where the lengths of documents are used as part of term weighting. It is normalization of tfidf-weighting.

Entropy weighting

$$w_{ij} = \log(f_{ij} + 1.0) * \left(1 + \frac{1}{\log(N)} \sum_{j=1}^{N}\left[\frac{f_{ij}}{df_j}\log\left(\frac{f_{ij}}{df_j}\right)\right]\right)$$

Where

$$\frac{1}{\log(N)} \sum_{j=1}^{N}\left[\frac{f_{ij}}{df_j}\log\left(\frac{f_{ij}}{df_j}\right)\right]$$

is average entropy of i^{th} term. The entropy of terms represents the degree to which the terms bear information content of document. It is minimum if i^{th} term occurs mostly equally in all documents, and maximum if the term occurs only in a few documents.

Annex B: naïve Bayesian model

Given a document *d*, it will estimate the condition probability P(c_i|*d*) for each class c_i, and *d* will be assigned to the class for which this condition probability is highest. P(c_i|*d*) can be computed by use of Bayes theorem as follows.

$$P(c_i|d) = \frac{P(d|c_i)P(c_i)}{P(d)}$$

Because the *P(d)* does not influence the resulting class with the highest $P(c_i|d)$, so the denominator of $P(c_i|d)$ can be ignored. The equation above can be rewritten as

$$P(c_i|d) = P(d|c_i)P(c_i).$$

The independence between terms means that

$$P(d|c_i) = P(c_i)\prod_{1}^{T} P(t_j|c_i).$$

Then $P(c_i)$ and $P(t_j|c_i)$ can be estimated as follows.

$$P(c_i) = \frac{|c_i|}{N} \quad \text{and} \quad P(t_j|c_i) = \frac{1 + \sum_{d_l \in c_i} f_{jl}}{T + \sum_{r=1}^{T} \sum_{d_l \in c_i} f_{rl}}$$

Here $|c_i|$ is the number of training document in the class c_i , T is the number of total terms in a optimal feature term subset and f_{jl} is the number of times j^{th} term occurs in the l^{th} training document d_l.