A Study on Document Classification using Machine Learning Techniques

Kabita Thaoroijam

Haldia Institute of Technology
West Bengal, India

Abstract

With the explosion of information fuelled by the growth of the World Wide Web it is no longer feasible for a human observer to understand all the data coming in or even classify it into categories. With this growth of information and simultaneous growth of available computing power automatic classification of data, particularly textual data, gains increasingly high importance. Text classification is a task of automatically sorting a set of documents into categories from a predefined set and is one of the important research issues in the field of text mining. This paper provides a review of generic text classification process, phases of that process and methods being used at each phase.

Keywords: *Machine learning algorithm, document representation, classification, performance evaluation.*

1. Introduction

Due to the fast growth of digital information available electronically, text mining plays a key role in managing information and knowledge, and therefore has become an active research area. Text mining, also known as intelligent text analysis is the process of extracting interesting and non-trivial information and knowledge from unstructured text. Text mining is a young interdisciplinary field, which draws on information retrieval, data mining, machine learning, statistics and computational linguistics. Typical text mining tasks include information extraction, topic tracking, document summarization, classification, clustering, question answering [1]. Automated text classification is the act of dividing a set of input documents into two or more classes where each document can be said to belong to one or multiple classes. Text classification aims at assigning pre-defined classes to text documents. An example would be to automatically label each incoming news story with a topic like "sports", "politics", or "art". The classification task starts with a training set $D = (d_1, ..., d_n)$ of documents that are already labelled with a class $c \in C$ (e.g. sport, politics). The task is then to determine a classification model

$$f : D \rightarrow C \qquad f(d) = c$$

which is able to assign the correct class to a new document d of the domain.

Text classification is a challenging task, as it is difficult to capture the meaning and abstract concepts of natural language just from a few keywords. Also, the high dimensionality of the feature space makes classification problem very difficult. Text classification is commonly used to handle spam emails, classify large text collections into topical categories, manage knowledge and also to help Internet search engines.

The rest of the paper is organized as follows. In Section 2 an overview of documents representation approaches is given, Section 3 presents dimension reduction approaches, Section 4 presents machine learning techniques for document classification, in Section 5 classifier evaluation method is presented and finally in section 6 conclusion is made.

2. Document Representations

Before any classification task, one of the most fundamental tasks that need to be accomplished is that of document representation and feature selection. The most commonly used document representation is the vector space model which was originally developed for automatic indexing [16]. Under the vector space model, a collection of n documents with m unique terms is represented as an $m \times n$ term-document matrix (where each document is a vector of m dimensions). Several terms weighing schemes have been used, including binary term frequency and simple term frequency (i.e. how many times the words occur in the document). In the most popular scheme, the document vectors are composed of weights reflecting the frequency of the terms in the document multiplied by the inverse of their frequency in the entire collection. ($tf \times idf$) .The assumption is that words which occur frequently in a

document but rarely in the entire collection are of highly discriminative power.

3. Dimension Reductions

A central problem in document classification is the high dimensionality of the feature space and a relatively small number of training samples. There exists a dimension for each unique word found in the document collection. Standard classification technique cannot deal with such a large feature set, and hence there is a need for reduction of the original feature set, which is known as dimension reduction. There are two approaches to dimension reduction: Feature Selection and Re-parameterization.

3.1 Feature Selection

Feature selection attempts to remove non-informative words from documents in order to improve classification effectiveness and reduce computational complexity. The main idea of feature selection is to select subset of features from the original documents by keeping the words with highest score according to predetermined measure of the importance of the word .The selected features retains original physical meaning and provide a better understanding for the data and learning process . A comparative study of different feature selection methods for document classification is reported in [21].

3.1.1 Document Frequency Thresholding

The document frequency for a word is the number of documents in which the word occurs. In document frequency thresholding one computes the document frequency for each word in the training corpus and removes those words whose document frequency is less than some predefined threshold. The basic assumption is that rare words are either non-informative for category prediction or not influential in global performance.

3.1.1 Information Gain

Information Gain measures the number of bits of information obtained for category prediction by knowing the presence or absence of a word in a document. Let $c_1,\ldots,c_k$ denote the set of possible categories. The information gain of a word w is defined as:

$$IG(w) = -\sum_{j=1}^{k} P(c_j)\log P(c_j) + P(w)\sum_{j=1}^{k} P(c_j|w)\log P(c_j|w) + P(\bar{w})\sum_{j=1}^{k} P(c_j|\bar{w})\log P(c_j|\bar{w})$$

Here $P(c_j)$ can be estimated from the fraction of documents in the total collection that belongs to class c_j and $P(w)$ from the fraction of documents in which the word w occurs. Moreover, $P(c_j \mid w)$ can be computed as the fraction of documents from class c_j that have at least one occurrence of word w and $P(c_j \mid \bar{w})$ as the fraction of documents from class c_j that does not contain word w.

The information gain is computed for each word of the training set, and the words whose information gain is less than some predefined threshold are removed.

3.1.1 Mutual Information

Mutual information (MI) measure is derived from information theory, and provides a formal way to model the mutual information between the features and the classes. MI measures how much information the presence/absence of a word w contributes to making the correct classification decision on c. Formally

$$MI(w,c) = \sum_{w\in\{0,1\}}\sum_{c\in\{0,1\}} P(w,c)\log\left(\frac{P(w,c)}{P(w)P(c)}\right)$$

Here $P(w,c)$ is the joint probability of w and c. MI measures how much information - in the information-theoretic sense - a term contains about the class. If a term's distribution is the same in the class as it is in the collection as whole, then MI is 0. MI reaches its maximum value if the term is a perfect indicator for class membership, that is, if the term is present in a document if and only if the document is in the class.

3.1.1 χ^2 Statistics

The χ^2 statistic measures the lack of independence between a word w and category c. Using the two-way contingency table of a word w and category c, where A is defined as number of times w and c co-occur. B is number of times w occurs without c. C is number of times c occurs without w. D is number of times either c or w occurs and N is the total number of documents. It is defined as

$$\chi^2(w,c) = \frac{N\times(AD-CB)^2}{(A+C)\times(B+D)\times(A+B)\times(C+D)}$$

The χ^2 statistic value is zero if w and c are independent.

3.6 Re-parameterization

Re-parameterization is the process of constructing new features as combinations or transformation of the original

features. One such approach is Latent Semantic Indexing LSI [18].This technique compresses document vectors into vectors of a lower-dimensional space whose dimensions are obtained as combinations of the original dimensions by looking at their patterns of co-occurrence. In practice, LSI infers the dependence among the original terms from a corpus and "wires" this dependence into the newly obtained, independent dimensions. The function mapping original vectors into new vectors is obtained by applying a singular value decomposition to the matrix formed by the original document vectors. In classification this technique is applied by deriving the mapping function from the training set and then applying it to training and test documents alike.

4. Machine Learning Techniques

After feature selection and transformation, the documents can be easily represented in a form that can be used by a machine learning algorithm. Although many approaches have been proposed, automated text classification is still a major area of research primarily because the effectiveness of current automated text classifiers is not faultless and still needs improvement. The various machine learning techniques for document classification have been studied in [4, 8].

4.1 Rocchio's Algorithm

Rocchio [14] is the classic method for document routing or filtering in Information retrieval. In this method, a prototype vector is built for each class c_j and a document vector d is classified by calculating the similarity between d and each of the prototype vectors, then assign document to the class with maximum similarity. The prototype for class c_j is computed as the average vector over all training document vectors that belongs to c_j. This means that learning is very fast for this method.

4.2 Naive Bayes Classifier

In probabilistic classifier [3], the probability that a d_i document represented by a vector $\langle w_1, w_2, ..., w_m \rangle$ of weighted terms belongs to $P(c_j)$, and this probability is computed by an application of Bayes' theorem, given by

$$P(c_j|d_i) = \frac{P(c_j)P(d_i|c_j)}{P(d_i)}$$

Here $P(d_i)$ is the probability that a randomly picked document has d_i vector as its representation, and $P(c_j)$ the probability that a randomly picked document belongs to c_j.

The naïve bayes classifer make the assumption that any words of the document vector are, when viewed as random variables, statistically independent of each other; this independence assumption is encoded by the equation

$$P(d_j|c_i) = P(c_j)\prod_{k=1}^{m} P(w_{kj}|c_i)$$

Despite the fact that the assumption of conditional independence is generally not true for word appearance, the naïve bayes classifier is surprisingly effective in documents.

4.3 k-nearest neigbour

To classify an unknown document vector d, the k nearest neighbor algorithm [10] ranks the document's neighbors among the training document vectors, and use the class labels of the k most similar neigbours to predict the class of the input document. The classes of these neigbours are weighted using the similarity of each neighbor to d, where similarity can be calculated by the cosine similarity measure. Given two documents d_1 and d_2 the cosine similarity, $\cos(\theta)$, is represented using a dot product and magnitude as

$$\cos(\theta) = \frac{d_1.d_2}{\|d_1\|\|d_2\|}$$

The k-NN is quite simple and effective but its drawbacks is its inefficiency at classification time: it requires the entire training set to be ranked for similarity with the test document which is expensive.

4.4 Decision Tree

Decision trees are one of the most widely used inductive learning methods. One of the most well-known decision tree algorithms is ID3 [12] and its successor C4.5 [13] and C5[5]. A decision tree is a flowchart like tree structure, where each internal node denotes a test on an attribute, each branch represents an outcome of the test, and each leaf node holds a class label. Given a document, for which the associated class label is unknown, the attribute values of the document are tested against the decision tree. A path is traced from the root to a leaf node, which holds the class prediction for that document. The tree starts as a single root node containing all of the training examples. If the training examples are all from the same class, then the node becomes a leaf, labeled with that class. Else, an attribute selection method is called to determine the splitting

criterion. Such a method may use a heuristic or statistical measure (e.g., information gain or gini index) to select the "best" way to separate the training examples into individual classes. Next, the node is labeled with the splitting criterion, which serves as a test at the node. A branch is grown from the node to each of the outcomes of the splitting criterion and the training examples are partitioned accordingly. The algorithm uses the same process recursively to create a decision tree for the training examples at each partition.

4.5 Neural Networks

Neural networks consist of many individual processing units called as neurons connected by links which have weights that allow neurons to activate other neurons. Each unit receives a set of inputs, which are denoted by the vector d_i, which corresponds to the term frequencies in the i^{th} document. Each neuron is also associated with a set of weights A, which are used in order to compute a function $f(\cdot)$ of its inputs. A typical function which is often used in the neural network is the linear function as follows:

$$p_i = A \cdot d_i$$

Thus, for a vector d_i drawn from a lexicon of m words, the weight vector A should also contain m elements. Now consider a binary classification problem, in which all labels are drawn from $\{+1, -1\}$. Assume that the class label of d_i is denoted by c_j. In that case, the sign of the predicted function p_i yields the class label. In order to illustrate this point, consider a simple example in a 2-dimensional feature space, as illustrated in Figure 1.

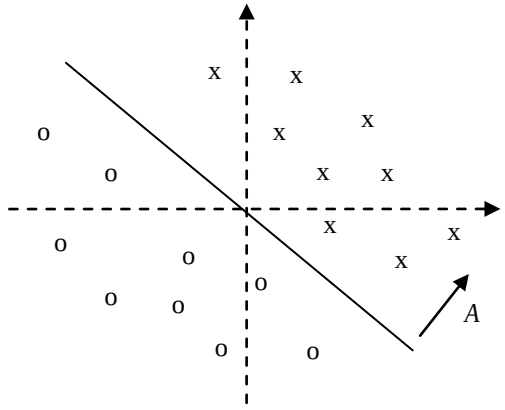

Figure1: The sign of the projection onto the weight vector A yields the class label

In this case, it is illustrated with two different classes, and the plane corresponding to $Ax = 0$ is illustrated in the same figure. It is evident that the sign of the function $A \cdot d_i$ yields the class label. Thus, the goal of the approach is to learn the set of weights A with the use of the training data. The idea is that you start off with random weights and gradually update them when a mistake is made by applying the current function on the training example. The magnitude of the update is regulated by a learning rate μ. This forms the core idea of the perceptron algorithm.

A number of implementations of neural network methods for text data have been studied in [2, 9, 11, 17, 20].

If all the classes may not be neatly separated from one another with a linear separator, multiple layers of neurons can be used in order to induce such non-linear classification boundaries. The effect of such multiple layers is to induce multiple piece-wise linear boundaries, which can be used to approximate enclosed regions belonging to a particular class. In such a network, the outputs of the neurons in the earlier layers feed into the neurons in the later layers. The training process of such networks is more complex, as the errors need to be back-propagated over different layers. Some examples of such classifiers include those discussed in [7, 15, 19, 22].

The advantage of the high flexibility of neural networks entails the disadvantage of very high computing costs. Another disadvantage is that neural networks are extremely difficult to understand for an average user; this may negatively influence the acceptance of these methods.

4.6 Support Vector Machine

A Support Vector Machine (SVM) algorithm introduced in text classification by [6] has been extensively and successfully used for text classification tasks. In geometrical terms, it may be seen as the attempt to find, among all the surfaces $\sigma_1, \sigma_2 \ldots$ in n dimensional space that separate the positive from the negative training examples (decision surfaces), the σ_i that separates the positives from the negatives by the widest possible margin, that is, such that the separation property is invariant with respect to the widest possible translations of σ_i.

This idea is best understood in the case in which the positives and the negatives are linearly separable, in which case the decision surfaces are $n-1$ hyperplanes. In the two-dimensional case of Figure 2, various lines may be chosen as decision surfaces. The SVM method chooses the middle element from the "widest" set of parallel lines, that is, from the set in which the maximum distance between two elements in the set is highest. It is noteworthy that this "best" decision surface is determined by only a small set of training examples,

called the support vectors. The method described is applicable also to the case in which the positives and the negatives are not linearly separable [22]. The most important property of SVMs is that learning is nearly independent of the dimensionality of the feature space. It rarely requires feature selection as it inherently selects data points (the support vectors) required for a good classification. This allows good generalization even in the presence of a large number of features and makes SVM especially suitable for the classification of texts.

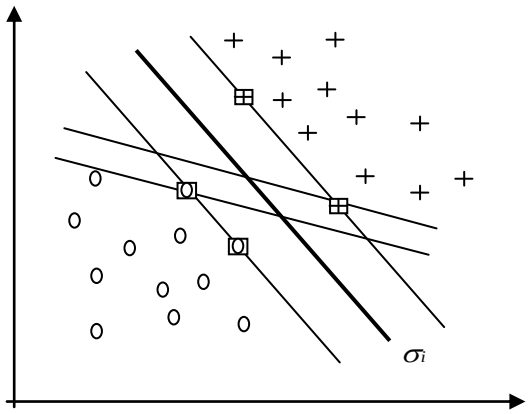

Figure 2 – Linear Support Vector Machine

In the figure the small crosses and circles represent positive and negative training examples, respectively, whereas lines represent decision surfaces. Decision surface σ_i (indicated by the thicker line) is, among those shown, the best possible one, as it is the middle element of the widest set of parallel decision surfaces (i.e., its minimum distance to any training example is maximum). Small boxes indicate the support vectors.

5. Evaluation Method

An important issue of document classification is how to measure the performance of the classifiers. Many measures have been used, each of which has been designed to evaluate some aspect of the classifier performance. Precision and recall are the most common measures for evaluating an information retrieval system. Precision is the proportion of returned documents that are targets, while recall is the proportion of target documents returned.

Table 1: Contingency table for category j

Category j		**Expert judgment**	
		True	**False**
Classifier Judgement	True	TP_j	FP_j
	False	FN_j	TN_j

Table 2: The Global Contingency table

Category set $C=\{c_1, c_2, \ldots, c_k\}$		**Expert judgment**	
		True	**False**
Classifier Judgement	True	$TP=\sum_{j=1}^{k} TP_j$	$FP=\sum_{j=1}^{k} FP_j$
	False	$FN=\sum_{j=1}^{k} FN_j$	$TN=\sum_{j=1}^{k} TN_j$

Formally,

$$P_j = \frac{TP_j}{TP_j + FP_j} \qquad R_j = \frac{TP_j}{TP_j + FN_j}$$

There are two conventional methods of calculating the performance of a document classification system based on precision and recall. The first is called micro-averaging, while the second one macro-averaging. Micro-averaged values are calculated by constructing a global contingency table and then calculating precision and recall using these sums. In contrast macro-averaged scores are calculated by first calculating precision and recall for each category and then taking the average of these. The notable difference between these to calculations is that micro-averaging gives equal weight to every document (it is called a document-pivoted measure) while macro-averaging gives equal weight to every category (category-pivoted measure).

$$P_{micro} = \frac{\sum_{j=1}^{k} TP_j}{\sum_{j=1}^{k} TP_j + FP_j} \qquad R_{micro} = \frac{\sum_{j=1}^{k} TP_j}{\sum_{j=1}^{k} TP_j + FN_j}$$

$$P_{macro} = \frac{1}{k}\sum_{j=1}^{k} \frac{TP_j}{TP_j + FP_j} \qquad R_{macro} = \frac{1}{k}\sum_{j=1}^{k} \frac{TP_j}{TP_j + FN_j}$$

Another evaluation criterion that combines precision and recall is the F-measure.

$$F = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

6. Conclusion

The classification problem is one of the most fundamental problems in the machine learning and data mining literature. Almost all the known techniques for classification such as decision trees, rules, Bayes

methods, nearest neighbour classifiers, SVM classifiers, and neural networks have been extended to the case of text data. Recently, a considerable amount of emphasis has been placed on linear classifiers such as neural networks and SVM classifiers, with the latter being particularly suited to the characteristics of text data. Process of text classification is well researched, but still many improvements can be made both to the feature preparation and to the classification engine itself to optimize the classification performance for a specific application. Research describing what adjustments should be made in specific situations is common, but a more generic framework is lacking.

References

[1] Andreas Hotho, Andreas Nürnberger, and Gerhard Paaß: **A brief survey of text mining**, In LDV Forum - GLDV Journal for Computational Linguistics and Language Technology 20(1):19-62, May 2005.

[2] Dagan I., Karov Y., Roth D.: **Mistake-driven Learning in Text Categorization,** In Proceedings of EMNLP, 1997.

[3] David D. Lewis: **Naive (Bayes) at Forty: The Independence Assumption in Information Retrieval,** In ECML '98 Proceedings of the 10th European Conference on Machine Learning, 1998, pp 4-15.

[4] Fabrizio Sebastiani: **Machine learning in automated text categorization,** In ACM Computing Surveys, 2002, 34(1):1-47.

[5] http://www.rulequest.com/see5-info.html

[6] Joachims, T.: **Text categorization with support vector machines: learning with many relevant features**, In Proceedings of ECML-98,10th European Conference on Machine Learning (Chemnitz, Germany, 1998), pp 137–142.

[7] Lam S., Lee D.: **Feature reduction for neural network based text categorization**, In DASFAA Conference, 1999.

[8] Li Y. H. and Jain A. K.: **Classification of Text Documents**, In The Computer Journal, Volume 41, Issue 8, pp. 537-546.

[9] Littlestone N.: **Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm**, In Machine Learning, 2: 1988, pp. 285–318.

[10] Mitchell, T.M.: **Machine Learning**, McGraw Hill, New York, NY, 1996..

[11] Ng H. T., Goh W., Low K.: **Feature selection, perceptron learning, and a usability case study for text categorization**, In ACM SIGIR Conference, 1997.

[12] Quinlan, J.: **Induction of Decision trees**, In Machine Learning, 1, 81-106, 1986.

[13] Quinlan, J.: **C4.5: programs for Machine Learning**, Morgan Kaufman San Matteo,CA, 1993.

[14] Rocchio, J.: **Relevance Feedback in Information Retrieval**, In G. Salton (ed.). The SMART System: pp.67-88.

[15] Ruiz M., Srinivasan P.: **Hierarchical neural networks for text categorization**, In ACM SIGIR Conference, 1999.

[16] Salton G., Wong A., and Yang C. S.: **A Vector Space Model for Automatic Indexing**, In Communications of the ACM, vol. 18, nr. 11, pages 613–620, 1975.

[17] Schutze H., Hull D., Pedersen J.: **A comparison of classifiers and document representations for the routing problem**, In ACM SIGIR Conference, 1995.

[18] Scott C. Deerwester, Susan T. Dumais, Thomas K. Landauer, George W. Furnas, Richard A. Harshman: **Indexing by Latent Semantic Analysis**, In Journal of the American society for information science 41(6): 391-407, 1990.

[19] Weigand A., Weiner E., Pedersen J.: **Exploiting hierarchy in text categorization**, In Information Retrieval, 1(3), pp. 193–216, 1999.

[20] Wiener E., Pedersen J. O., Weigend A. S.: **A Neural Network Approach to Topic Spotting**, In SDAIR, pp. 317–332, 1995.

[21] Yang, Y., Pedersen J.P.: **A Comparative Study on Feature Selection in Text Categorization**, In Proceedings of the 14th International Conference on Machine Learning (ICML'97), pp. 412-420, 1997.

[22] Yang Y., Liu L.: **A re-examination of text categorization methods**, In ACM SIGIR Conference, 1999.