RESEARCH ARTICLE OPEN ACCESS

Discussion on Different Algorithm for Text Classification and Feature Selection

Nilophar Mulani, Yoginath Kalshetty
Department of Computer Science& Engg. College of engg. Pandharpur
Department of Computer Science & engg. College of engg. Pandharpur

Abstract
In different application domains as well as areas of research text classification is one of the well studied problems. So there is need to enhance the effective and efficient algorithm for text classification .There are many algorithm presented by different authors over the successfully and accurate text classification by different researchers. Each algorithm presented are specific to applications or some other domains of research. Some techniques presented are based on data mining and machine learning domains. The main aim of this paper is to summarize the different types of algorithm presented for text classification. In this paper we have presented the key components for text classification which will be helpful for researcher to understand the existing techniques of text classification. First we will give the overview of why there is need for feature reduction and different technique for feature selection, then the key components of text classification system. Later we will discuss the different algorithm of text classification.
Keywords: feature clustering, Text Classification, Feature selection

I. Introduction

As the data grows in size day by day,there is big demand for text classification. So that required data can be accessed easily.Text classification is a supervised learning task for assigning text document to pre-defined classes of document. It is used to find valuable information from huge collection of text document available in digital libraries, knowledge database, the World Wide Web. Artificial intelligence provides many learning methods and paradigm to represent, interpret and acquire domain knowledge to help other document. As the dataset has huge size and complexity ,data dimensionality reduction has become a primary need before text classification. Feature reduction can be used to reduce the dimensionality of feature vector .there are three ways for feature reduction. Feature selection and feature extraction and feature clustering .feature clustering is one of the powerful method for feature reduction. Text classification with fuzzy logic provides a better forum to categorize the text and web document. When it combined with feature clustering technique it highly improves the result.some of application in this field are sensitive text classification technique, cyber terrorism investigation, spam filtering .because the unlabeled data is easy to store but not helpful as the labeled data.

Some key methods, which are commonly used for feature selection are described in section 2 .we will describe decision tree methods for text classification in section 3. Rule based classifiers are described in detail in section 4. We discuss naïve Bayes classifiers in section 5.. In section 6, we will discuss SVM classifier. In section 7 we will conclude.

II. Feature Selection for Text Classification

Before any classification task, one of the most fundamental tasks that needs to be accomplished is that of document representation and feature selection. While feature selection is also desirable in other classification tasks, it is especially important in text classification due to the high Dimensionality of text features and the existence of irrelevant (noisy) features. In general, text can be represented in two separate ways. The first is as a bag of words, in which a document is represented as a set of words, together with their associated frequency in the document. Such a representation is essentially independent of the sequence of words in the collection. The second method is to represent text directly as *strings*, in which each document is a sequence of words. Most text classification methods use the bag-of-words representation because of its simplicity for classification purposes. In this section, we will discuss some of the methods which are used for feature selection in text classification. The most common feature selection which is used in both supervised and unsupervised applications is that of stop-word removal and stemming. In stop-word removal, we determine the common words in the documents which are not specific or discriminatory to the different classes. In stemming, different forms of the same word are consolidated into a single word. For example, singular, plural and different tenses are consolidated into a single word. We note that these methods are not specific to the case of the classification problem, and are often used in a variety of unsupervised applications such as clustering and

indexing. In the case of the classification problem, it makes sense to supervise the feature selection process with the use of the class labels. This kind of selection process ensures that those features which are highly skewed towards the presence of a particular class label are picked for the learning process. A wide variety of feature selection methods are discussed in [133, 135]. Many of these feature selection methods have been compared with one

2.1 Gini Index

One of the most common methods for quantifying the discrimination level of a feature is the use of a measure known as the *gini-index*. Let $p1(w)$ . . . $pk(w)$ be the fraction of class-label presence of the k different classes for the word w. In other words, $pi(w)$ is the conditional probability that a document belongs to class i, given the fact that it contains the word w.

2.2 Information Gain

Another related measure which is commonly used for text feature selection is that of information gain or entropy. Let Pi be the global probability of class i, and $pi(w)$ be the probability of class i, given that document contains the word w.

Let $F(w)$ be the fraction of the documents containing the word w. The information gain measure $I(w)$ for a given word w is defined as follows:

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

$$P(w)\sum_{j=1}^{K} P(c_j \mid w)\log P(c_j \mid w)$$

$$+ P(\bar{w})\sum_{j=1}^{K} P(c_j \mid \bar{w})\log P(c_j \mid \bar{w})$$

2.3 Mutual Information

This mutual information measure is derived from information theory , and provides a formal way to model the mutual information between the features and the classes. The point wise mutual information $Mi(w)$ between the word w and the class i is defined on the basis of the level of co-occurrence between the class i and word w. We note that the expected co-occurrence of class i and word w on the basis of mutual independence is given by Pi . Either of these measures may be used in order to determine the relevance of the word w. The second measure is particularly useful, when it is more important to determine high levels of positive correlation of the word w with any of the classes.

2.4 χ2-Statistic

The χ2 statistic is a different way to compute the lack of independence between the word w and a particular class i. Let n be the total number of documents in the collection, $pi(w)$ be the conditional probability of class i for documents which contain w, Pi be the global fraction of documents containing the class i, and $F(w)$ be the global fraction of documents which contain the word w .As in the case of the mutual information, we can compute a global χ2 statistic from the class-specific values. We can use either the average of maximum values in order to create the composite value.We note that the χ2-statistic and mutual information are different ways of measuring the correlation between terms and categories. One major advantage of the χ2-statistic over the mutual information measure is that it is a normalized value, and therefore these values are more comparable across terms in the same category.

III. Decision Tree Classifiers

A decision tree [6] is essentially a hierarchical decomposition of the (training) data space, in which a *predicate* or a condition on the attribute value is used in order to divide the data space hierarchically. In the context of text data, such predicates are typically conditions on the presence or absence of one or more words in the document. The division of the data space is performed recursively in the decision tree, until the leaf nodes contain a certain minimum number of records, or some conditions on class purity. The majority class label (or cost-weighted majority label) in the leaf node is used for the purposes of classification. For a given test instance, we apply the sequence of predicates at the nodes, in order to traverse a path of the tree in top-down fashion and determine the relevant leaf node. In order to further reduce the over fitting, some of the nodes may be be pruned by holding out a part of the data, which are not used to construct the tree. The portion of the data which is held out is used in order to determine whether or not the constructed leaf node should be pruned or not. In particular, if the class distribution in the training data (for decision tree construction) is very different from the class distribution in the training data which is used for pruning, then it is assumed that the node overfits the training data. Such a node can b pruned. In the particular case of text data, the predicates for the decision tree nodes are typically defined in terms of the underlying text collection.

IV. Rule-based Classifiers

Decision trees are also generally related to *rule-based classifiers*. In rule-based classifiers, the data space is modeled with a set of rules, in which the left hand side is a condition on the underlying feature set, and the right hand side is the class label. The rule set is essentially the model which is generated from the training data. For a given test instance, we determine the set of rules for which the test instance satisfies the condition on the left hand side of the rule. We determine the predict class label as a function of the class labels of the rules which are satisfied by the test instance. In its most general form, the left hand side of the rule is a boolean condition, which is

expressed in Disjunctive Normal Form (DNF). However, in most cases, the condition on the left hand side is much simpler and represents a set of terms, all of which must be present in the document for the condition to be satisfied. The *absence* of terms is rarely used, because such rules are not likely to be very informative for sparse text data, in which most words in the lexicon will typically not be present in it by default (sparseness property). Also, while the *set intersection* of conditions on term presence is used often, the union of such conditions is rarely used in a single rule. This is because such rules can be split into two separate rules, each of which is more informative on its own. We note that decision trees and decision rules both tend to encode rules on the feature space, except that the decision tree tends to achieve this goal with a hierarchical approach. In fact, the original work on decision tree construction in C4.5 [6] studied the decision tree problem and decision rule problem within a single framework. This is because a particular path in the decision tree can be considered a rule for classification of the text instance. The main difference is that the decision tree framework is a strict hierarchical partitioning of the data space, whereas rule-based classifiers allow for overlaps in the decision space. The general principle is to create a rule set, such that all points in the decision space are covered by *at least* one rule. In most cases, this is achieved by generating a set of targeted rules which are related to the different classes, and one default *catch-all* rule, which can cover all the remaining instances. A number of criteria can be used in order to generate the rules from the training data. Two of the most common conditions which are used for rule generation are those of *support* and *confidence*. These conditions are common to all rule-based pattern classifiers [88] and may be defined as follows:

Support: This quantifies the **absolute number** of instances in the training data set which are relevant to the rule. For example, in a corpus containing 100,000 documents, a rule in which **both** the left-hand set and right-hand side are satisfied by 50,000 documents is more important than a rule which is satisfied by 20 documents.

Essentially, this quantifies the statistical *volume* which is associated with the rule. However, it does not encode the strength of the rule.

Confidence: This quantifies the **conditional probability** that the right hand side of the rule is satisfied, if the left-hand side is satisfied. This is a more direct measure of the strength of the underlying rule. We note that the afore-mentioned measures are not the only measures which are possible, but are widely used in the data mining and machine learning literature [12] for both textual and non-textual data, because of their intuitive nature and simplicity of interpretation. One criticism of the above measures is that they do not normalize for the a-priori presence of different terms and features, and are therefore prone to misinterpretation, when the feature distribution or class-distribution in the underlying data set is skewed. The training phase constructs all the rules, which are based on measures such as the above. For a given test instance, we determine all the rules which are relevant to the test instance. Since we allow overlaps, it is possible that more than one rule may be relevant to the test instance.

If the class labels on the right hand sides of all these rules are the same, then it is easy to pick this class as the relevant label for the test instance.

On the other hand, the problem becomes more challenging when there are conflicts between these different rules. A variety of different methods are used to rank-order the different rules [12], and report the most relevant rule as a function of these different rules. For example, a common approach is to rank-order the rules by their confidence, and pick the top-*k* rules as the most relevant. The class label on the right-hand side of the most number of these rules is reported as the relevant one. Aim interesting rule-based classifier for the case of text data has been proposed in [13]. This technique uses an iterative methodology, which was first proposed in [14] for generating rules. Specifically, the method determines the single best rule related to any particular class in the training data. The best rule is defined in terms of the confidence of the rule, as defined above. This rule along with its corresponding instances are removed from the training data set. This approach is continuously repeated, until it is no longer possible to find strong rules in the training data, and complete predictive value is achieved.The transformation of decision trees to rule-based classifiers is discussed generally in [6], and for the particular case of text data in [15].

For each path in the decision tree a rule can be generated, which represents the conjunction of the predicates along that path. One advantage
of the rule-based classifier over a decision tree is that it is not restricted to a strict hierarchical partitioning of the feature space, and it allows for overlaps and inconsistencies among the different rules. Therefore, if a new set of training examples are encountered, which are related to a new class or new part of the feature space, then it is relatively easy to modify the rule set for these new examples. Furthermore, rule-based classifiers also allow for a tremendous interpretability of the underlying decision space.

V. Probabilistic and Naive Bayes Classifiers

Probabilistic classifiers are designed to use an implicit mixture model for generation of the underlying documents. This mixture model typically assumes that each class is a component of the mixture. Each mixture component is essentially a generative model, which provides the probability of sampling a particular term for that component or

class. This is why this kind of classifiers are often also called generative classifier. The naive Bayes classifier is perhaps the simplest and also the most commonly used generative classifers. It models the distribution of the documents in each class using a probabilistic model with independence assumptions about the distributions of different terms. Two classes of models are commonly used for naive Bayes classification. Both models essentially compute the posterior probability of a class, based on the distribution of the words in the document. These models ignore the actual position of the words in the document, and work with the "bag of words" assumption. The major difference between these two models the assumption in terms of taking (or not taking) word frequencies into account, and the corresponding approach for sampling the probability space:

Multivariate Bernoulli Model:

In this model, we use the presence or absence of words in a text document as features to represent a document. Thus, the frequencies of the words are not used for the modeling a document, and the word features in the text are assumed to be binary, with the two values indicating presence or absence of a word in text. Since the features to be modeled are binary, the model for documents in each class is a multivariate Bernoulli model.

Multinomial Model:

In this model, we captuer the frequencies of terms in a document by representing a document with a bag of words. The documents in each class can then be modeled as samples drawn from a multinomial word distribution. As a result, the conditional probability of a document given a class is simply a product of the probability of each observed word in the corresponding class.

No matter how we model the documents in each class (be it a multivariate Bernoulli model or a multinomial model), the component class models (i.e., generative models for documents I each class) can be used in conjunction with the Bayes rule to compute the posterior probability of the class for a given document, and the class with the highest posterior probability can then be assigned to the document. There has been considerable confusion in the literature on the differences between the multivariate Bernoulli model and the multinomial model. In the following, we describe these two models in more detail.

5.1 Bernoulli Multivariate Model

This class of techniques treats a document as a set of distinct words with no frequency information, in which an element (term) may be either present or absent. The seminal work on this approach may be found in [82]. Let us assume that the lexicon from which the terms are drawn are denoted by $V = \{t1 \ldots tn\}$. Let us assume that the bag-of-words (or text document) in question contains the terms $Q = \{ti1 \ldots tim\}$, and the class is drawn from $\{1 \ldots k\}$. Then, our goal is to model the posterior probability that the document (which is assumed to be generated from the term distributions of one of the classes) belongs to class i, given that it contains the terms $Q = \{ti1 \ldots tim\}$. The best way to understand the Bayes method is by understanding it as a sampling/generative process from the underlying mixture model of classes. The Bayes probability of class i can be modeled by sampling a set of terms T from the term distribution of the classes: If we sampled a term set T of any size **from the term distribution of one of the randomly chosen classes**, and the final outcome is the set Q, then what is the posterior probability that we had originally picked class i for sampling? The a-priori probability of picking class i is equal to its fractional presence in the collection. We denote the class of the sampled set T by CT and the corresponding posterior probability by $P(CT = i|T = Q)$. This is essentially what we are trying to find. It is important to note that since we do not allow replacement, we are essentially picking a subset of terms from V with no frequencies attached to the picked terms. Therefore, the set Q may not contain duplicate elements. Under the naive Bayes assumption of independence between terms, this is essentially equivalent to either selecting or not selecting each term with a probability that depends upon the underlying term distribution. Furthermore, it is also important to note that this model has no restriction on the number of terms picked.

VI. SVM classifiers

Support-vector machines were first proposed for numerical data. The main principle of SVMs is to determine separators in the search space which can best separate the different classes. Support vector machines (SVMs) have been recognized as one of the most successful classification and computational complexity of training in support vector machines may be independent of the dimension of the feature space, reducing computational complexity is an essential issue to efficiently handle a large number of terms in practical applications of text classification. In this paper, we adopt novel dimension reduction methods to reduce the dimension of the document vectors dramatically. We also introduce decision functions for the centroid-based classification algorithm and support vector classifiers to handle the classification problem where a document may belong to multiple classes. Our substantial experimental results show that with several dimension reduction methods that are designed particularly for clustered data, higher efficiency for both training and testing can be achieved without sacrificing prediction accuracy of text classification .

VII. Conclusions

The classification problem is one of the most fundamental problems in the machine learning and data mining literature. In the context of text data, the problem can also be considered similar to that of classification of *discrete set-valued* attributes, when the frequencies of the words are ignored. The domains of these sets are rather large, as it comprises the entire lexicon. Therefore, text mining techniques need to be designed to effectively manage large numbers of elements with varying frequencies. Almost all the known techniques for classification such as decision trees, rules, Bayes methods, nearest neighbor 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. In recent years, the advancement of web and social network technologies have lead to a tremendous interest in the classification of text documents containing links or other meta-information. Recent research has shown that the incorporation of linkage information into the classification process can significantly improve the quality of the underlying results.

Refences

[1] T. M. Cover, J. A. Thomas. *Elements of information theory*. New York: John Wiley and Sons, 1991.

[2] Y. Yang, J. O. Pederson. A comparative study on feature selection in text categorization

[3] Y. Yang. Noise Reduction in a Statistical Approach to Text Categorization.

[4] D. Lewis, M. Ringuette. A comparison of two learning algorithms for text categorization. *SDAIR*, 1994.

[5] S. Chakrabarti, S. Roy, M. Soundalgekar. Fast and Accurate Text Classification via Multiple Linear Discriminant Projections, *VLDB Journal*, 12(2), pp. 172–185, 2003.

[6] T. Joachims. Text categorization with support vector machines: learning with many relevant features.

[7] J. R. Quinlan, Induction of Decision Trees, *Machine Learning*,

[8] Y. Li, A. Jain. Classification of text documents. *The Computer Journal*, 41(8), pp. 537–546, 1998.

[9] schapire, Y. Singer. BOOSTEXTER: A Boosting-based System for Text Categorization, *Machine Learning*, 39(2/3), pp. 135–168, 2000.

[10] S. Dumais, J. Platt, D. Heckerman, M. Sahami. Inductive learning algorithms and representations for text categorization. *CIKM Conference*, 1998.

[11] D. Chickering, D. Heckerman, C. Meek. A Bayesian approach for learning Bayesian networks with local structure. *Thirteenth Conference on Uncertainty in Artificial Intelligence*, 1997.

[12] B. Liu, W. Hsu, Y. Ma. Integrating Classification and Association Rule Mining. *ACM KDD Conference*, 1998.

[13] B. Liu, W. Hsu, Y. Ma. Integrating Classification and Association Rule Mining. *ACM KDD Conference*, 1998.

[14] S. M. Weiss, N. Indurkhya. Optimized Rule Induction, *IEEE Exp.*, 8(6), pp. 61–69, 1993.

[15] D. Johnson, F. Oles, T. Zhang, T. Goetz. A Decision Tree-based Symbolic Rule Induction System for Text Categorization, *IBM Systems Journal*, 41(3), pp. 428–437, 2002.

[16] W. Cohen, Y. Singer. Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems*, 17(2), pp. 141–173, 1999.

[17] W. Cohen. Learning rules that classify e-mail. *AAAI Conference*, 1996.

[18] W. Cohen. Learning with set-valued features. *AAAI Conference*,1996.

[19] Y. Freund, R. Schapire, Y. Singer, M. Warmuth. Using and combining predictors that specialize.