Apply Support Vector Machine to Text Classification

Yang Zhang (11529139)

I. Abstract

This report analyzed the feasibility of classifying texts using Support Vector Machine. In this report, I will explain the reasons why SVM is suitable to solve the text categorization problem by exploring the properties of text classification problem. In the experiment part, empirical data are provided to support the theoretical ideas.

II. Introduction

Text classification or document categorization is a common problem in information science and computer science. The task is to assign a document to one or more classes or categories. The media not only limits to plain text, but also is expendable to images, sound tracks, etc. With the rapid growth of online information, text classification is increasingly important and widely used in search engine and media databases. People are already benefited from this technology, for example, they can read news by their interests without manually filtering. It will be advantageous to train a robust classifier from samples.

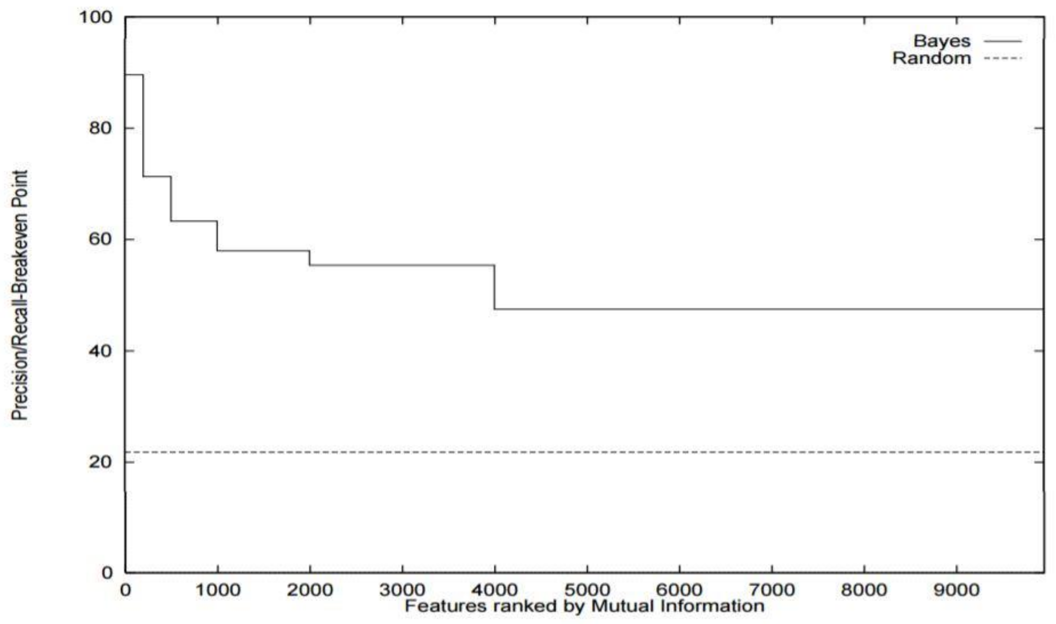
Naïve Bayes is a popular algorithm that commonly used for solving text classification problem. However, Naïve Bayes has its own limitation that cannot be easily overcome and there are noticeable benefits of applying support vector machines in this problem. (I will discuss this detailly in next part of the report). The approach of proving the feasibility of SVM is to compare the performance between Naïve Bayes and SVM on the same set of classification tasks. The result will be explored in the experiment part.

III. Problem Setup and Theoretical Supports

The main task of text classification is to assign a document with one or more predefined tags. In the matter of solving the problem by machine learning, the objective is to automatically learn classifiers from provided training examples. This is a kind of supervised learning problem, which SVM can be fitted in with general speculation. In fact, if thinking deeper, SVM should work well in text classification problem. To support the hypothesis, let’s see how merits of SVM integrate into the properties of general text classification.

Firstly, the features vector of text set is very large. The size could be more than dozens of thousands. While, such high dimensional input space is not a problem to SVM. Due to it not depends on the number of features, SVM is able to deal with such a feature vector with many features.

Secondly, there are only a few irrelevant features. The figure below shows the results of an experiment on the Reuters “acq” category. [1] In the graph, all features are ranked based on their frequency.



The plot shows the accuracy of Naïve Bayes classifier that is trained by using features ranked 1-200, 201-500, 501-1000, 1001-2000, 2001-4000, 4001-9962. We can clearly observe that the classifier trained even with low ranking features has much better accuracy than the classifier trained with randomly selected feature. This results suggest that the number of redundant features are small, if we try to avoid high dimensional input space through aggressive feature selection, we may also train a underfitting classifier because of huge loss of information.

Thirdly, the feature vectors in text classification are sparse. As I seen in hw4 and the vector generated from my experiment, the size of feature vectors could be large (a few thousand attributes), however there are only a few of entries are non-zero. This is because the texts even with same topic are barely overlapped. In P.Auer’ paper *The perceptron algorithm vs. winnow: Linear vs. logarithmic mistake bounds when few input variables are relevant* [2], he proved that Additive kind of algorithms work well for solving problems with sparse samples. His opinion suggests that SVM is suitable for text classification.

From the three points, we can say that in theory, to use SVM solving text classification problem is reasonable.

IV. Solution Approach and Experiments

The idea of this experiment is to compare the performance of SVM with the commonly used text classification algorithm Naïve Bayes on the same test settings.

1. Preparation

To construct the experiments, 2 things are required to be done in advance. There are 1. Data collection. 2. Algorithm Software

1. Data Preparation
   I found 3 datasets from technion [3]
   (1) r52-test-no-stop (2568 samples) r52-train-no-stop (6532 samples)
   (2) webkb-test-stemmed (1396 samples) webkb-train-stemmed (2803)
   (3) r8-test-no-stop (2189 samples) r8-train-no-stop (5485 samples)
2. Algorithm Software
   (1) Weka [4] for experimenting with Naïve Bayes
   (2) Libsvm [5] for experimenting with SVM

2. Experiment Procedure

The first step is always to twit with the data, in order word, to make them useable. The good thing about the data sets I found is the author already filtered out all the common stop words (such as 'the', 'a', 'an', etc.), I need to do is to convert them to feature data that meet the specific requirements.
To make datasets fit in both Weka and Libsvm, a feature transfer tool (written in Java) is used to convert each document into following formats:

1. CSV format file for Weka
2. <lable> <index>:<feature> format file for Libsvm

After having the transferred data, using the feature data of training set to build Naïve Bayes and SVM model, for the SVM model, it was trained with 3 polynomial kernel (polynomial degree = 1, 2, 3). Last, using the feature data of testing set to measure the performance (accuracy) of each model

3. Experiment Results

Dataset	NB	SVM (d=1)	SVM(d=2)	SVM(d=3)
webkb-train-stemmed	66.1	71.4	75.4	76.3
webkb-test-stemmed	68.8	66.9	68.5	69.4
r8-train-no-stop	61.2	59.3	65.1	67.6
r8-test-no-stop	58	61.8	64.3	65.9
r52-train-no-stop	65	75.1	75.3	75.6
r52-test-no-stop	62.6	73.1	74.5	74.2

For the Naïve Bayes classifier, the average accuracy over the 3 test sets is 65%.

For the SVM, the average accuracy over the 3 test sets is 69%.

From the accuracy stats, we can see that the SVM performs better than Naïve Bayes. However, when I was doing the experiment, the training time of SVM is noticeable longer than Naïve Bayes. This is not harder to understand, the training process of SVM is a typical quadratic programming problem, which is more expensive than the linear programming in Naïve Bayes.

V. Conclusions

From the theoretical discuss, we acknowledge the advantage of applying SVM to solve text classification problem. (1) SVM can deal with large input space feature vector. (2) SVM works well in the case that the feature vectors are sparse.

The experimental results show that the SVM has reasonable good performance compared to Naïve Bayes. And the classifier trained by SVM is more robust. Over the 3 test sets, SVM performed stable in all cases, while Naïve Bayes shows spike in r8 data set.

As a conclusion, even though the training process of SVM is expensive compared to Naïve Bayes, SVM is still a good alternative method for text classification, which can provide relatively good and stable performance.

VI. Acknowledgments and References

Team member: Yang Zhang

1. T. Joachims, *A probabilistic analysis of the rocchio algorithm with tfidf for text categorization.*
2. P.Auer, M. Warmuth, J. Kivinen. *The perceptron algorithm vs. window: Linear vs. logarithmic mistake bounds when few input variables are relevant.*
3. Data source from *Israel Institute of Technology*
4. Weka is open source ware from University of Waikato: *http://www.cs.waikato.ac.nz/ml/weka/*
5. Libsvm is open source library from Chih-Chung Chang and Chih-Jen Lin: http://www.csie.ntu.edu.tw/~cjlin/libsvm/