uOttawa

L'Université canadienne
Canada's university

FACULTÉ DES ÉTUDES SUPÉRIEURES ET POSTOCTORALES

uOttawa

L'Université canadienne
Canada's university

FACULTY OF GRADUATE AND POSDOCTORAL STUDIES

Yimin Ma

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M.C.S.

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Text Classification on Imbalanced Data: Application to Systematic Reviews Automation

TITRE DE LA THÈSE / TITLE OF THESIS

Stan Matwin

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

CO-DIRECTEUR (CO-DIRECTRICE) DE LA THÈSE / THESIS CO-SUPERVISOR

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

D. Inkpen (teleconference)

F. Oppacher

Gary W. Slater

Le Doyen de la Faculté des études supérieures et postdoctorales / Dean of the Faculty of Graduate and Postdoctoral Studies

Text classification on imbalanced data: Application to Systematic Reviews Automation

Yimin Ma

Thesis submitted to
the faculty of graduate and postdoctoral studies
in partial fulfillment of the requirements
for the degree of Master of Computer Science (MCS)

School of Information Technology and Engineering
Faculty of Engineering
University of Ottawa

Your file Votre référence
ISBN: 978-0-494-34085-1
Our file Notre référence
ISBN: 978-0-494-34085-1

Contents

List of Tables

List of Figures

Abstract

Systematic Review is the basic process of Evidence-based Medicine, and consequently there is urgent need for tools assisting and eventually automating a large part of this process. In the traditional Systematic Review System, reviewers or domain experts manually classify literatures into relevant class and irrelevant class through a series of systematic review levels. In our work with TrialStat, we apply text classification techniques to a Systematic Review System in order to minimize the human efforts in identifying relevant literatures. In most cases, the relevant articles are a small portion of the Medline corpus. The first essential issue for this task is achieving high recall for those relevant articles. We also face two technical challenges: handling imbalanced data, and reducing the size of the labeled training set.

To address these issues, we first study the feature selection and sample selection bias caused by the skewness data. We then experimented with different feature selection, sample selection, and classification methods to find the ones that can properly handle these problems. In order to minimize the labeled training set size, we also experimented with the active learning techniques. Active learning selects the most informative instances to be labeled, so that the required training examples are reduced while the performance is guaranteed. By using an active learning technique, we saved 86% of the effort required to label the training examples. The best testing result was obtained by combining the feature selection method *Modified BNS*, the sample selection method *clustering-based sample selection* and active learning with the Naïve Bayes as classifier. We achieved 100% recall for the minority class with the overall accuracy of 58.43%. By achieving work saved over sampling (WSS) as 53.4%, we saved half of the workload for the reviewers.

Acknowledge

I would like to express my deepest and sincere gratitude to my supervisor, Professor Stan Matwin, for all the help, inspiration, advice and guidance I have had from him during my master's study at the University of Ottawa. Professor Matwin has always been an enthusiastic advisor who gave me lots of creative ideas when I was facing technical difficulties. His enthusiasm, excitement, persistence, and professionalism at researching for innovative text classification solutions have always been motivating me to continuously expand my knowledge in this field.

I would like to thank Professor Diana Inkpen and William Elazmeh for bringing in many fresh ideas and constructive suggestions during the project meetings, and for giving me invaluable feedbacks after proof-reading the drafts of this thesis.

I would like to thank the University of Ottawa, the Natural Sciences and Engineering Research Council (NSERC) and Ontario Graduate Scholarship Program (OGS) for their financial support.

Last and most importantly, I would like to thank my husband and mother for their encouragement, support, and help. Without that, this thesis would not have been possible.

Chapter 1

Introduction

1.1 Background

A Systematic Reviews is a literature review for identifying high quality research evidence relevant to a specific research topic [Pai et al., 2004]. It is one of the critical processes in delivery of health care. In the traditional Systematic Review Systems, reviewers or domain experts manually classify literature items into relevant class and irrelevant class through a series of systematic review levels. In the first level, the title, abstracts and keywords of a document are reviewed to determine if it is relevant to the research topic. In the second level, the full text will be reviewed to judge if it meets the inclusion criteria. In the remaining levels, information that makes a document pass the previous two levels will be extracted. This labor-intensive process is slow and expensive; therefore, we want to use a text classification system to automate most of the jobs. The first two levels of the systematic review system are typical two-class text classification tasks. With some labeled documents as training data, we can train a text classifier to classify the unlabeled documents into relevant or irrelevant class.

The benefits of applying text classification in a Systematic Review System are:

1. Reduce the number of documents that need to be labeled manually to save the cost and time.
2. Help reviewers to prioritize the documents, so they can choose the important documents to label when the time is limited
3. Reduce the chance of human error.

In this thesis, we apply text classification techniques to assist an existing Systematic Review System, developed by an Ottawa company called TrialStat, to reduce the number of literature items that need to be manually classified by domain experts.

We have some challenges when applying text classification on Systematic Review Systems. Traditionally, the text classification uses a set of labeled documents of n classes as training data to build a classifier, and then uses this classifier to categorize the unlabeled documents into the n classes. Most of the supervised learning algorithms for text classification assume that the class distribution in the training set is reasonably balanced, but it is not the case in Systematic Review. The reason is that the literature items provided to a Systematic Review System are retrieved from a large database or electronic library based on the set of acquired keywords. However, articles in different topics can share the same key words, which often cause only a small proportion of literature items in the query result set to be relevant to the topic of interest. Viewed from the classification perspective, the data is imbalanced: the relevant articles are vastly outnumbered by the ones that are not relevant. Therefore, the text classification system that assists Systematic Review should be able to handle the imbalanced data problem.

The second challenge is that the overhead workload of deploying a text classification is large, due to the large human effort to label the training examples. Manually labeling training documents is expensive and time consuming. Especially for some scientific technical or medical literatures, domain experts are required to perform the job.

The final challenge is the fact that the cost of missing relevant literature is very high in Systematic Review. In most cases, however, relevant literatures are only a small portion of the whole data. When the data is imbalanced, most of the text classification algorithms pay more attention to the majority class to achieve high accuracy, which results in low recall for the minority class.

The target of this thesis is to find the text classification techniques that can achieve high recall for the relevant class with reasonable precision. We first study the feature selection

and sample selection bias caused by the skewness of the data. We then experiment with different feature selection, sample selection, and classification methods to find the ones that can properly handle these problems. In order to minimize the labeled training set size, we also experimented with the active learning techniques. Active learning selects the most informative instances to be labeled, so that the required training examples are reduced while the performance is guaranteed.

In this thesis, documents are represented as bag-of-words. In the bag-of-words approach, each distinct word in the document is considered as a feature. These features could be represented as the absolute term frequency, which is the number of times this word occurs in the document; or a binary value, which indicates whether this word is present in the document or not. In this way, a document can be represented as a vector of numeric values, and the dimensions of this vector (coordinates) are referred as the feature space.

1.2 Main contributions

The first contribution of this thesis is the experimentation with different feature selection methods, under-sampling methods and classification methods in the high skewed data situation. As we discussed before, most of the text classification techniques assume the class distribution in the training set is reasonably balanced. Therefore, their behavior will change if they are applied in the imbalanced data situation, and bias will occur. So we have to test the adaptability of different techniques to the skewed data set.

The second contribution is the exploration of the relations between the optimal feature ratio and the class distribution. Optimal feature ratio changes when the class distribution and the importance of a class change. We found that the smoothed class distribution function proposed in [Tang and Liu, 2005] works well in predicting the optimal feature ratio.

The third contribution is the introduction of a new sample selection method, *clustering based sample selection*, to select unlabeled examples to be labeled. In our study, this method is used to form the initial training set for the active learning system. When the data is highly imbalanced, this method is able to get sufficient minority class examples for training without labeling a large number of examples. This method first clusters the unlabeled examples into two clusters, and then requests the labels of the examples in the center of these clusters. By using this method, the noisy and redundant examples can be reduced. Furthermore, we saved 70% of the labeled examples.

The last contribution is the experimentation with the active learning techniques for text classification when class distribution is highly skewed. The main idea of active learning is to select the examples with the largest uncertainty to be labeled. We use active learning techniques to improve the classification performance by introducing more informative examples into the training set.

Chapter 2

Related Work

This chapter presents a brief introduction to the text classification algorithms and feature selection methods that are used in this thesis. In the end of this chapter, some previous works on classifying the imbalanced data will be discussed.

2.1 Text classification algorithms

In this section, we will present the text classification algorithms we used in this thesis, which include Naïve Bayes, SVM and Decision Tree. These algorithms are very popular in text classification because of their solid mathematical foundation and ability to handle the high dimensionality of the feature space. Due to the simplicity, computational efficiency and interpretability of Naïve Bayes, it is used as a base learner in this thesis.

2.1.1 Naïve Bayes

Naïve Bays is a well-known probabilistic algorithm, which is based on the Bayes' theorem of conditional probability. Let $C = \{c_1, c_2, \ldots c_n\}$ be the set of possible labels for the documents. Each document d_i is represented as a vector of term frequency X, where $X = \{x_{i1}, x_{i2}, \ldots x_{im}\}$. The term frequency x_{im} is how many times the given term m appears in the document d_i. According to the Bayes' rule, the posterior probability for label c_j for a given document d_i would be:

$$P(c_j|d_i) = \frac{P(d_i|c_j)\,P(c_j)}{P(d_i)} = \frac{P(x_{i1}, x_{i2}, \ldots x_{im}|\, c_j)\,P(c_j)}{P(X)} \qquad (2.1)$$

P(X) can be ignored, since it is the same for all the classes and it does not depend on the class. The joint probability in the numerator can be decomposed as follow:

$$\begin{aligned} & P(x_{i1}, x_{i2}, \ldots x_{im}|c_j) \\ &= P(x_{i1}|c_j)\, P(x_{i2}, \ldots x_{im}|c_j, x_{i1}) \\ &= P(x_{i1}|c_j)\, P(x_{i2}|c_j, x_{i1})\, P(x_{i3}, \ldots x_{im}|c_j, x_{i1}, x_{i2}) \\ &= P(x_{i1}|c_j)\, P(x_{i2}|c_j, x_{i1})\, P(x_{i3}|c_j, x_{i1}, x_{i2})\, P(x_{i4}, \ldots x_{im}|c_j, x_{i1}, x_{i2}, x_{i3}) \\ &= \ldots \end{aligned} \tag{2.2}$$

In order to simplify the computation, Naïve Bayes makes an assumption that the attributes are independent in a given class. Since in text classification an attribute is represented by the number of times a given word occurs in a document, assuming the independence of attributes implies assuming the occurrence of each word in the document is independent. The joint probability can be simplified to be the product of the probability of each attribute in a given class c.

Assume all the attributes are independent, $P(x_{ip}|c_j, x_{iq}) = P(x_{ip}|c_j)$, while x_{ip} is not equal to x_{iq}. The joint probability can be simplified to:

$$\begin{aligned} & P(x_{i1}, x_{i2}, \ldots x_{im}|c_j) \\ &= P(x_{i1}|c_j)\, P(x_{i2}|c_j)\, P(x_{i3}|c_j)\, P(x_{i4}|c_j) \ \ldots\ P(x_{im}|c_j) \end{aligned} \tag{2.3}$$

The formula of the Naïve Bayes becomes:

$$P(c_j|X) = P(c_j) \prod_{k=1}^{m} P(x_{ik}|\, c_j) \tag{2.4}$$

Although the Naïve Bayes assumption is very easy to violate in the real world, it still performs very well in many domains [Lewis and Ringuette, 1994; Craven et al., 1998; Yang & Pederson, 1997; Joachims, 1997]. In [Friedman 1997], the author explains that, the classifier is optimal under zero-one loss even when this assumption is violated by a wide margin. That is because classification estimation is only a function of the sign of the function estimation.

There are two popular Naïve Bayes approaches that are used in text classification, multi-variate Bernoulli and multinomial [McCallum and Nigam, 1998b]. Multi-variate Bernoulli represents a document by a vector of binary attributes where each attribute indicates whether a word is present in this document or not. This approach treats the probability of a document as a collection of independent Bernoulli experiments for each word in the vocabulary. By using the Bernoulli approach, the document probability in formula (2.4) could be replaced by

$$P(x_{ik}|c_j) = \prod_{k=1}^{m}(B_{ik}P(w_{ik}|c_j)) + (1 - B_{ik})(1 - P(w_{ik}|c_j)) \qquad (2.5)$$

where B_{ik} is 1 if the word w_k occurs in document d_i; otherwise, B_{ik} is 0. $P(w_{ik}|c_j)$ is the probability of word w_k appears in all the documents with class c_j. The Bernoulli approach uses all the words in the vocabulary to calculate the document probability.

The Multinomial approach represents a document by a vector of word frequencies, where each attribute indicates the number of times a word occurs in this document. In this approach, the document probability is a drawn from a multinomial distribution of words in vocabulary with m independent trials, where m equals the number of words in the document. This approach assumes the length of a document is independent of the class. By using the Multinomial approach, the document probability in formula (2.4) could be replaced by

$$P(x_{ik}|c_j) = P(|d_i|)|d_i|! \prod_{k=1}^{m} \frac{P(w_{ik}|c_j)^{N_{ik}}}{N_{ik}!} \qquad (2.6)$$

where N_{ik} is the number of times word w_k occurs in document d_i, and $|d_i|$ is the length of document d_i. The Multinomial approach only uses the words that occur in the document to calculate the document probability.

According to the experimental result in [McCallum and Nigam, 1998b], Bernoulli outperforms Multinomial when the number of attributes is small, while Multinomial outperforms Bernoulli when the number of attributes is large. In our studies, the number

of attributes is reduced by feature selection to a relatively small pool, so the Bernoulli is more suitable for our experiments.

In [Rennie et al., 2003], the authors modified the Multinomial Naïve Bayes to solve two systemic errors, skewed data bias and feature independent assumption, that affect Naïve Bayes performance. This method is called Complement Naïve Bayes. Complement Naïve Bayes solves the imbalanced training data problem by using the data from all other classes except class c to estimate whether the document belongs to class c. A document will be assigned to class c when its probability of being in classes other than c is low (refer to formula 2.7). In the two-class classification, this method uses the negative data to estimate the likelihood of the positive class, and vice versa. The classification rule for Complement Naïve Bayes is as follows:

$$l_{CNB}(d_i) = \text{argmax}_c \left[\log P(c) - \sum P(w_k) \log \frac{N_{\bar{c}k} + \alpha_k}{N_{\bar{c}} + \alpha}\right] \qquad (2.7)$$

where $P(c)$ is the probability of the occurrence of class c, $P(w_k)$ is the number of times the word k occurs in document d_i, $N_{\bar{c}k}$ is the total number of times word k occurs in classes other than c, and N_c is the total number of words in classes other than class c. α_k and α are the smoothing parameters. In [Rennie et al., 2003], α_k equals to 1 and α equals to the sum of the α_k (i.e. α is equal to the number of words in the bag-of-words representation used).

Complement Naïve Bayes defines the weight for word k in class c as,

$$w_{ck} = \log \frac{N_{\bar{c}k} + \alpha_k}{N_{\bar{c}} + \alpha} = \log \theta_{ck} \qquad (2.8)$$

Refer to the second component of formula 2.7, the product of $P(w_k)$ and $\log \theta_{ck}$ will be summed to estimate the label for the document d_i. For example, we want to classify articles into two categories: articles about preventing West Nile and articles about preventing SARS, and assume the words "West" and "Nile" always appear together. By using formula 2.7 to estimate the label for a document, when the term "West Nile"

occurs, each word in the term will be counted once, so this term will be double counted; on the other hand, when the word "SARS" occurs once, it will only be counted once. Therefore, the dependent feature has larger contribution to the final result than the independent feature when their numbers of occurrence are the same.

Complement Naïve Bayes solves the feature independence assumption problem by normalizing the weight vector.

$$w_{ck} = \frac{\log\theta_{ck}}{\sum_m |\log\theta_{cm}|} \tag{2.9}$$

Where m is the total number of words occurring in class c.

2.1.2 Support Vector Machine

A Support vector machine (SVM) is a supervised learning method for classification and regression. Developed by Vladimir Vapnik in 1982, SVM rapidly gained the attention of the machine learning community due to its strong theoretical foundations. It has been successfully applied to different areas such as bioinformatics, pattern recognition and text classification. Suppose the data is considered as geometrical points in an n-dimension space, the basic idea of SVM is to find hyperplanes that can linearly separate these points. SVM is used for the two-class classification initially, but it can be extended to handle the multi-class classification. In this thesis only the two-class classification is applied.

Assume instances in training data are linearly separable in input space, there exists a set of hyperplanes that can correctly separate instances into two classes. We want to find the optimal hyperplane that gives the greatest separation between the two classes, so that the distance between the closest data points to the hyperplane is maximized. This optimal hyperplane is called *maximum margin hyperplane*. The data points with minimum distance to the hyperplane are called *support vectors*, and the maximum distance between the support vectors and the hyperplane is called the *margin*. The decision boundary of

SVM is relatively stable, because the maximum margin hyperplane would not change unless support vectors are added or deleted. This characteristic of SVM prevents the occurrence of overfitting.

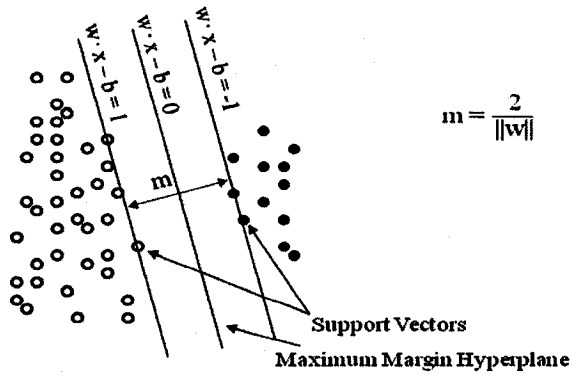


Figure 2.1 Geographic representation of the support vectors and the maximum margin hyperplane

Denote the training data as $\{x_i, c_i\}$, $i = 1, 2, \ldots n$, a hyperplane can be described as

$$\Sigma w_i x_i - b = 0 \quad => \quad w \cdot x - b = 0$$

where $x_i \subseteq R^N$, $c_i \subseteq \{1, -1\}$, b is the offset, and w_i is the weight to be learned. The class label of an instance is the sum of all the weights multiplied by the instance value, which is equal to the inner product of the weights and the instance values. The decision function can be described as $f(x) = \text{sign}(w \cdot x - b)$. Taking two instances x_1 and x_2 from the two classes $(w \cdot x_1) - b = 1$ and $(w \cdot x_2) - b = -1$ respectively, the margin of these points is the distance between these points, and the *maximized margin hyperplane* can be found by minimizing the value of w.

$$\frac{w}{||w||} \bullet (x_1 - x_2) = \frac{2}{||w||}$$

If the input data are not linearly separable in the input space, SVM uses the kernel function to convert them into a high-dimensional space, called feature space, so that there is a better chance the data are linearly separable.

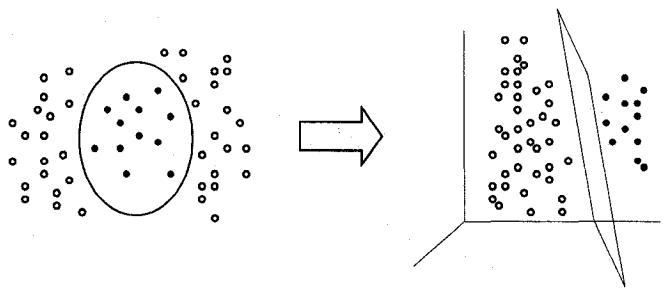


Figure 2.2 How the non-linearly separable instances become separable in higher dimension

The data in input space can be projected to feature space F so that it is linearly separable. The projection can be defined as Φ: $R^N \rightarrow F$, where the linearly separable classification is performed in *F*. The feature space has more dimensions and data more likely could be separated there, however, the calculation becomes very complex.

For example, project the data from two-dimensional space to three-dimensional space, Φ: $R^2 \rightarrow R^3$:

$$\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \Rightarrow \begin{pmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{pmatrix}$$

The dot product of the two vectors projected to F becomes:

$$\begin{pmatrix} x_1^2 \\ x_1 x_2 \\ x_2^2 \end{pmatrix} \cdot \begin{pmatrix} y_1^2 \\ y_1 y_2 \\ y_2^2 \end{pmatrix} = x_1^2 y_1^2 + x_1 x_2 y_1 y_2 + x_2^2 y_2^2$$

To simplify the calculation, the dot product in feature space can be replaced by the kernel function in the input space, $\Phi(x) \cdot \Phi(y) = k(x, y)$. For example, using polynomial function as the kernel function, $k(x, y) = (x \cdot y)^2$, we can prove that the dot product of the two vectors in feature space can be mapped to the kernel function in the input space.

$$\begin{pmatrix} x_1^2 \\ x_1x_2 \\ x_2^2 \end{pmatrix} \cdot \begin{pmatrix} y_1^2 \\ y_1y_2 \\ y_2^2 \end{pmatrix} = x_1^2y_1^2 + x_1x_2y_1y_2 + x_2^2y_2^2$$

$$= (x_1y_1 + x_2y_2)^2 = \left[\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \right]^2 = (\mathrm{x} \bullet \mathrm{y})^2$$

Some common kernel functions include:

- Polynomial: $k(\mathrm{x}, \mathrm{y}) = ((x \cdot y) + \theta)^d$
- Radial Basis Function: $k(\mathrm{x}, \mathrm{y}) = \exp(-\gamma \|x \cdot y\|^2)$, for $\gamma > 0$
- Sigmoid: $k(\mathrm{x}, \mathrm{y}) = \tanh(\kappa \mathrm{x} \cdot \mathrm{y} + c)$, for $\kappa > 0, c > 0$

However, when there are some noisy instances in the training data, the data could not be linearly separable in feature space. In this case, a slack-variable ξ is introduced to relax the hard-margin constraint.

$$c_i (w \cdot x_i - b) \geq 1 - \xi_i, i = 1, 2, \ldots n \qquad (2.10)$$

The *maximized margin hyperplane* could be obtained by minimizing $\|w\|$ and ξ at the same time.

$$\frac{1}{2} \|w\|^2 + C \sum_{i=1}^{N} \xi_i \qquad (2.11)$$

where $\xi_i \geq 0$ and C is used to balance the trade-off between the empirical error and the complexity term.

2.1.3 Decision Tree

Decision tree is a predictive model that shows the paths to reach the decision. In decision tree, each interior node does testing on an attribute, and each leaf gives a label for all instances that reach that leaf. To construct a decision tree, we can choose an attribute to test in the node and split the instances into subtrees according to the result of the test. This process can be performed recursively until all the instances in a node have the same

label. The value of an attribute could be either numeric or nominal. If an attribute contains a numeric value, the test in the node would compare the attribute value with a constant. If the attribute value is greater than or equal to this value, put the instance into a subtree; otherwise, put the instance into another subtree. The test in the node would also compare the attribute value with intervals so that the instances could be separated into more than two subsets. On the other hand, an attribute could contain a nominal value, for example, the weather could be sunny, cloudy, rainy... In this case, we can construct as many subtrees as the number of the possible value of the attribute. If only two subtrees are desired, we can classify the instances by checking if it has a specific value, for example, checking if the weather is sunny or not. The following diagram shows an example of the decision tree:

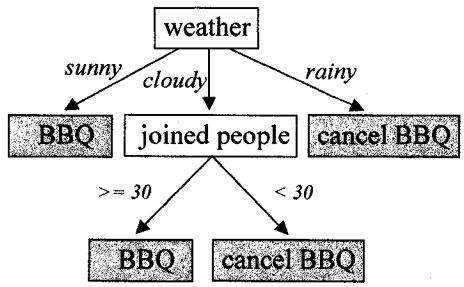


Figure 2.3 Decision Tree example

The most important point in constructing a decision tree is choosing a good attribute to test in the node. In order to reach the decision as soon as possible, we desire to have a short tree. We want to choose an attribute that could maximize the separation between the instances, so that the set of the instances in the subtree could be as pure as possible. The effectiveness of an attribute in classifying the instances can be measured by information gain. Let S be a set of training examples with labels C, where $C \subseteq \{c_1, c_2 \ldots c_n\}$, the *information* for this set can be defined as:

$$\text{Info}(S) = \sum_{i=1}^{n} - \left(\frac{f(c_i, S)}{|S|} \right) * \log_2 \left(\frac{f(c_i, S)}{|S|} \right) \qquad (2.12)$$

where $f(c_i, S)$ is the number of instances in S that with label c_i, and $|S|$ is the total number of instances in S. For the case that all instances belong to one class, we define $0\log_2(0)$

equals to 0. The *information* represents the amount of information that needs to be sent for classification. The value of the information is between 0 and 1. For example, if all instances belong to one class, the *information* equals to 0, which means no information is needed for classification. On the other hand, if instances are equally divided into two classes, the *information* equals to 1. Given an attribute A, decision tree induction splits training instances according to the value of A. The information gain tells how much i*nformation* is reduced after splitting the examples using A. The information gain can be defined as:

$$\text{Gain}(S, A) = \text{Info}(S) - \sum_{v \in \text{Values}(A)} \frac{|S_v|}{|S|} \text{Info}(S_v) \qquad (2.13)$$

where S_v is the subset of instances that the value for attribute A is v, $|S_v|$ is the number of instances in S_v, and value(A) is the set of all possible values of attribute A. The first term in the formula is the original *information* of the set S. The second term is the sum of the *information* of each subset weighted by the proportion of this subset in S.

Decision trees are trained to stop splitting the training data when they are perfectly classified. This behavior would easily cause the problem of overfitting, which means a more complex tree produces a larger error rate. Decision trees are overfitted when the training data is not generalized; for example, there exists noisy and correlative examples in the training set. Two approaches are commonly used to solve this problem, stop splitting the tree when data split is not statistically significant, or fully grow the tree but post-prune it afterwards. In [Lehnert et al., 1995], the authors suggest that adjusting the pruning thresholds can change the emphasis of recall or precision. Moderate pruning and branching levels tends to have the best precision at high recall levels, while large pruning and large branching tends to have best precision at low recall.

2.2 Feature selection methods

In this thesis, some popular features selection methods applied in text classification are studied in order to find the most suitable one for the imbalanced data situation. A brief introduction of these methods will be presented in the following section.

2.2.1 Document Frequency

Document Frequency is the number of documents that a word appears in. In this method, the words with document frequency greater than a predefined threshold will be selected. This criterion is based on the assumption that the words with low document frequency are less informative and have less influence to the global performance [Yang and Pedersen, 1997]. Furthermore, the words with high document frequency have better chance to appear in the testing data again. However, these high document frequency words do not include stop words[1]. Most of the stop words have high document frequencies; however, they do not have any contribution to the classification performance since they cannot reflect the topic of the articles. In our experiment, the stop words are removed before perform the feature selection. The advantage of document frequency is its computation efficiency and the knowledge of the class labels is not needed. Collection frequency is the total number of occurrences of a word in all documents. In our study, a document is represented by a vector of binary attributes where each attribute indicates whether a word is present in this document or not. Therefore, the Collection frequency is equal to Document Frequency in our study.

2.2.2 Information Gain

The information gain measures the loss of information when a term is absent in predicting class. Let us denote the categories as c_i where $i = 1, 2 \ldots m$. The information gain for a term t is defined as

1 Stop words are the non-informative words that appear very often in the document, such as prepositions, conjunctions, auxiliary verbs etc.

$$G(t) = -\sum_{i=1}^{m} P(c_i) \log P(c_i) + P(t) \sum_{i=1}^{m} P(c_i|t) \log P(c_i|t) + P(\overline{t}) \sum_{i=1}^{m} P(c_i|\overline{t}) \log P(c_i|\overline{t}) \quad (2.14)$$

where $P(c_i|t)$ is the probability that category c_i occurs given the term t and $P(c_i|\overline{t})$ is the probability that category c_i occurs given the terms other than t. When selecting a feature using this method, the words with information gain exceeding a pre-defined threshold will be selected.

2.2.3 Chi-square

Chi-square is a statistical test to measure the association between a feature and a class. Let t be a feature, c_i be a category where $i = 1, 2 \ldots m$, and N be the total number of documents, the association between c and t can be expressed as

$$\chi^2(t, c) = \frac{N * (P(t, c) * P(\overline{t}, \overline{c}) - P(\overline{t}, c) * P(t, \overline{c}))^2}{P(t) * P(\overline{t}) * P(c) * P(\overline{c})} \quad (2.15)$$

where $P(t, c)$ is the probability that both term t and category c occurs, $P(\overline{t}, \overline{c})$ is the probability that neither term t nor category c occurs, $P(\overline{t}, c)$ is the probability that category c occurs without term t and $P(t, \overline{c})$ is the probability that term t occurs without category c. If t and c are independent, the value of $\chi^2(t, c)$ is equal to zero. Compute $\chi^2(t, c_i)$ for a term t with each category c_i independently, and the Chi-square value for each term t is the average value of the associations between t and each category c_i, as shown in formula 2.14.

$$\chi^2(t) = \underset{k=1}{\overset{m}{avg}}\{\chi^2(t, c_i)\} \quad (2.16)$$

2.2.4 Odds Ratio

Odds ratio indicates the odds of the word occurring in the positive class normalized by that of the negative class [Forman, 2003]. In two-class classification scenario, let t be a feature, c_+ be the positive class and c_- be the negative class, the odds ratio for term t equals to

$$\text{Odds}(t) = \frac{P(t|c_+) * (1 - P(t|c_-))}{(1 - P(t|c_+)) * P(t|c_-)} \qquad (2.17)$$

This method favors the features that occur frequently in positive documents but seldom in negative documents. In [Mladenic and Grobelnik, 1999], this method was shown to outperform the other feature selection methods when it is used in combination with Multinomial Naïve Bayes.

2.3 Works on text classification on imbalanced data

Many real-life text classification tasks face the problem of highly skewed data. Since there is a trade-off between high precision and high recall, one application can only focus on one of these targets. For example, obtaining high precision is essential to search engines; while having high recall is important to automated article review systems. In this section, we will first review the related works on classifying research articles. Then we will look at different approaches used in the recent studies on addressing class imbalance.

2.3.1 Works on automated article retrieving systems

[Bartling, et al., 2003] present a method to retrieve the dental and craniofacial research literature from MEDLINE. The first stage in this method is to search MeSH[2] for terms

2 MeSH stands for Medical Subject Headings, it's a hierarchical system of key words or indexing terms assigns to each article to facilitate the retrieval in the medical literature database, MEDLINE.

related to stomatology, and then use these terms to acquire articles in MEDLINE. In the second stage, they randomly sampled the articles obtained in the first stage and had sixteen dental research experts classify these articles based only on the title and abstracts. Finally, they use these labeled articles to train a statistical text classification system, and find the distinct characteristics for the dental research papers according to the result. They tested the system on 990 articles, where 60% were dental research papers. The recall and precision for this system is 0.64 and 0.71 respectively.

In [Hu et al., 2005], the authors proposed a rule-based system for finding phosphorylation papers and extracting phosphorylation objects. This system uses shallow parsing and extracts phosphorylation information by matching text with manually developed patterns. The main architecture of this system includes sentence extraction and part of speech tagging, entity recognition, phrase detection, and relation identification. The main feature of this system performs information extraction, however, it still has an excellent performance in retrieving related papers, for example recall 96.4 and precision 91.4 with reasonable balanced class distribution 0.3.

[Zheng et al., 2006] proposed a cluster-based system to classify biomedical documents. In this system, negative training documents were first clustered into clusters according to their term distribution. Then a classifier was build for each cluster by taking the examples from the cluster as the negative examples and keep the whole positive set as positive examples. Finally, these clusters were used together to classify the test set. A document is classified as positive if and only if it is not classified as negative in all the classifiers. They test this system using four highly imbalanced datasets, which have skewness from 6% to 0.6%. The average result in these four experiments is 0.76 in recall and 0.23 in precision.

[Cohen et al., 2006] established an automated document classification system for classifying topic-specific evidence-based drug or therapy reviews. In classification system, the Chi-square is used as the feature selection method and the voting perceptron is used as the classifier. In their experiment, title, abstract, MeSH, MEDLINE

publication type for each article are used to generate the feature set. The proposed system was tested on the reference files from 15 systematic drug class reviews, where the skewness of the datasets varies from 0.5% to 27%. The experimental results show that this system reduced the number of articles needing manually review in 11 reviews, where the reduction rate for 3 of them was greater 50%.

2.3.2 Related works on addressing data imbalance

Different approaches are used to handle the skewed data problem in recent studies. We can divide them into three main categories: adjust the balance of the training data, adjust the feature ratio, and adjust the classifier to handle the imbalanced data problem.

The most intuitive way to solve the imbalanced data problem is to balance the training data by using the re-sampling techniques. In [Kubat and Matwin, 1997], the authors applied a directed under-sampling method to get rid of the noisy and redundant examples in the majority class. In [Japkowicz, 2000], the author compared several common re-sampling strategies, such as over-sampling, under-sampling and learning by recognition. According to her study, both over-sampling the minority class and under-sampling the majority class are effective ways to handle the imbalanced problem. She also found that under-sampling outperforms over-sampling on many domains. In [Alexandersson et al., 2005], different re-sampling methods such as under-sampling, over-sampling, boosting and bagging were used to re-balance the class distribution, and the under-sampling was found to perform the best. In [Drummond and Holte, 2003], the authors studied the impact of under-sampling and over-sampling on decision trees, and reported that under-sampling has better result. Similar study are performed in [Ling and Li, 1998] and [Domingos, 1999].

Another approach is to address the data bias problem in the feature selection. In [Forman, 2003], the author proposed a new feature selection method, Bi-Normal Separation (BNS), to handle the data imbalance problem. This feature selection method selects features from the positive class and the negative class proportional to the

smoothed class distribution. The detail of this method will be discussed in section 4.2.2. BNS was found to outperform the other feature selection methods for highly skewed data in [Forman, 2003] and [Tang and Liu 2005]. [Zheng et al., 2006] proposed a feature selection framework to select features from the minority class and the majority class separately by using the feature selection methods $\Im(t, c)$: Odds Ratio, Signed information Gain, or Correlation Coefficient are used as $\Im(t, c)$. Based on the fact that these feature selection methods tend to select the features representing the minority class, the authors assumed that larger $\Im$ value, the more likely feature t belongs to category c. They selected the features with highest $\Im(t, c_{minority})$ value to represent the minority class and the ones with lowest $\Im(t, c_{minority})$ value to represent the majority class. They empirically choose the best ratio between minority class features and majority class features. In this paper, a minority class feature is defined as the feature appearing only in the minority class, and a majority class feature is the one that only appears in the majority class.

People also spend lots of efforts in addressing the classifier to handle the data imbalance problem. One-class SVM is one of the examples. In [Manevitz and Yousef, 2001], the authors extended the SVM to handle the situation where only positive examples are available in the training data. This method can also apply to the case when the negative examples are hard to collect or the negative examples in the training set cannot represent the whole negative class. Also in [Zhuang and Dai, 2006] the authors proposed a general framework for one-class SVM which was shown to perform better than the standard one-class SVM. This framework contains three stages: training stage, estimation stage and adjustment stage. In the training stage, the one-class SVM classifier is trained from the minority instances with random initial parameters. In the estimation stage, both minority and majority instances are used to evaluate the performance of the classifier and estimate generalization performance measurement. In the adjustment stage, the parameter C of one-class SVM, which is used to balance the empirical error and complexity term, is tuned based on generalization performance measurement obtained from the second stage [Refer to formula 2.11]. This framework is tested on four UCI datasets with skewness varying from 1:6 to 1:130.

Besides SVM, Naïve Bayes was also modified to handle the imbalanced data problem. The Complement Naïve Bayes discussed in section 2.1.1 is one of the examples. This method was found to outperform the Multinomial Naïve Bayes, and performed as well as SVM on the Industry Sector corpus and the 20 Newsgroup corpus. In [Frank and Bouckaert, 2006] the authors analyzed the problem of Laplace correction in Naïve Bayes when the class is unbalanced, which will decrease the estimated probability of the minority class. They suggested solving this problem by normalizing the word count in each class so that the size of the class is the same for both classes after normalization. This method was tested on Reuters-21578, WebKB, Industry Section and 20 Newsgroup. It was found to outperform the Multinomial Naïve Bayes in most cases.

Chapter 3

The Data Sets

The data in this thesis are a set of clinical research papers provided by TrialStat Corporation. These papers are extracted by the librarians based on some given keywords. In their original Systematic Review System, the research papers are classified into *relevant* class and *irrelevant* class[3] through a series of systematic review levels. In the first level, the reviewers determine if the articles are appropriate for study based on their title, abstracts and keywords only. In the second level, reviewers evaluate the articles based on the full text. In the third level, reviewers extract information from articles that passed the two previous levels. More levels of screening may be performed depending on the different requirements of the customers. In each review level, the reviewer will be given some questions to answer. If the answer is positive or unsure, the paper will be passed to the next level of study. In order to ensure the accuracy, each article will be reviewed by at least two different reviewers.

Since it takes a long time to go through the whole article to answer the review questions in the second level, we try to filter out as many irrelevant documents as possible in the first level. Reading abstracts is faster and easier than reading the full text. However, if there are more than 48,000 articles in the dataset, how much time we would need to read all the abstracts? Suppose a person can read 100 abstracts and answer the related review questions per days, it would take 480 days. In this thesis, we try to automate the first level reviewing process by our text classification system. The input of the system is the metadata of the articles, such as the titles, abstracts or keywords.

---

3 The papers in *relevant* class are the ones that related to research topic, and papers in *irrelevant* class are the ones that not related to research topic.

This set of clinical research papers is about the Nutrition and Diets for Preventing and Healing Heart Disease and Stroke. After filtering out the articles without title or abstracts, there are 14,276 papers in this dataset. According to the label of the first level review, there are 551 relevant papers and 13725 irrelevant papers. The class distribution in this dataset is 3.8% in the relevant class and 96.2% in irrelevant class. This set of papers consists of 38,038 unique words. The average length for title and abstract is 13 and 250 words respectively. This corpus will be referred as corpus "Nutrition" thereafter.

The statistics for these datasets are as follows:

	Nutrition
Documents	14,276
Words	3,767,175
Unique words	38,038
Words in title	186,381
Unique words in title	10,929
Avg words per title	13
Words in abstract	3,580,794
Unique words in abstract	37,266
Avg words per abstract	250

Table 3.1 Statistics for the datasets

Chapter 4

The system overview and learning algorithms

The objective of this thesis is to find a solution for classifying the highly skewed text data, and achieve high recall in the minority class with small number of labeled training examples. There are two main issues in our studies, the first one is to handle the imbalanced data set, and the second one is to reduce the size of the labeled training set. The first issue is very important, because the imbalanced data could lead to some bias problems in feature selection, sample selection and classification. These bias problems need to be addressed in order to achieve good performance. Many techniques are introduced to handle these problems, and most of them contribute certain degree of improvement on the classification performance. These techniques are usually being experimented individually; however, there exist strong relations between them. As suggested by [Maloof, 2003], the class distribution in the training set, the class prior probability, the misclassification cost in each class, and the placement of the decision threshold are strongly connected. Modifying any one of them can affect the others. For example, changing the class distribution in the training set could change the decision thresholds of the classifier. The work in [Tang and Liu, 2005] shows that feature selection and over-sampling affect the performance of Naïve Bayes, Decision Tree and Support Vector Machines (SVM) in various degrees. In this thesis, we divide the text classification task into three stages: feature selection, sample selection and classification. We study several techniques to address different bias problems in different stages, and select the most suitable algorithms for achieving high recall. We also try to combine these algorithms to experiment with the influence between these techniques.

For second issue, we applied the active learning technique to minimize the size of the labeled training set. Active learning selects the most informative instances to be labeled, so that the required training examples are reduced while the performance is guaranteed. It is used in the situation in which obtaining labeled examples is difficult or expensive.

4.1 Text Classification System overview

A text classification system is developed as part of this thesis. This system could be divided into three parts: feature selection, sample selection and classification. We take the plain text as the input of the system. In feature selection stage, the plain text is converted into bag-of-words (BOW), which indicates whether this word is present in the document or not, and some of the important terms in the vector are selected to represent the documents. When going to the sample selection stage, the training set is formed. In this stage the re-sampling technique is used to balance the class distribution in the training set, and the active learning technique is applied to select the more informative examples to be labeled so that the required labeled samples is minimized. Finally, in the classification stage, the documents are classified into relevant class or irrelevant class. The architecture of the proposed system is presented in Figure 4.1.

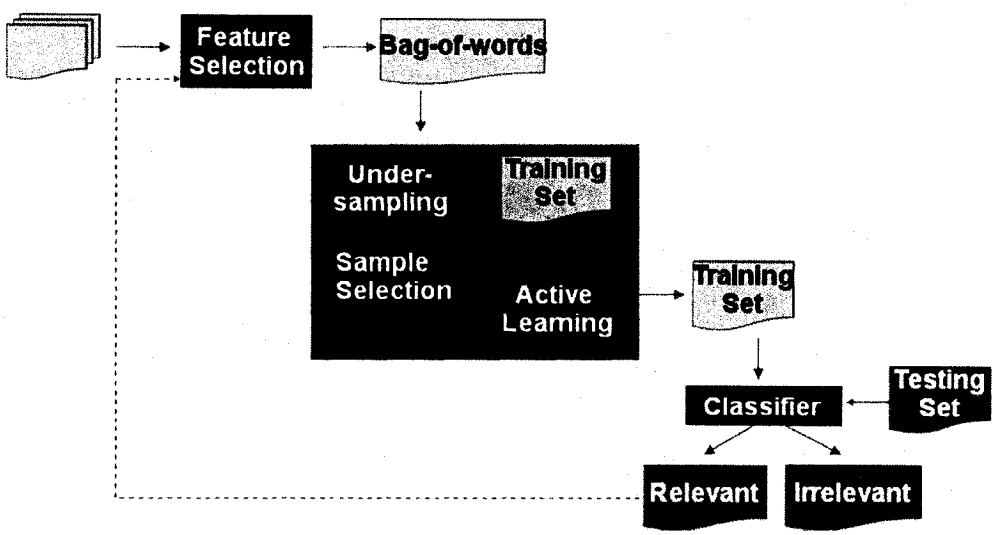


Figure 4.1Text Classification System overview

In the following sections, we will present the theory of all the algorithms that are used in each stage, and discuss the justification for selecting these algorithms. We will present the experimental results to support our justification in Chapter 5.

4.2 Feature Selection algorithms

One of the challenges in text classification is the high dimensionality of the feature space. With a few hundred of training documents, the feature set can easily contain thousands of features. The large feature set not only increases computational time but also increases the size of training data. Feature selection can improve the classification efficiency by reducing the dimensionality of the feature space by eliminating the irrelevant features and redundant features. By the definition, the irrelevant features are the features that do not contribute to the predictive accuracy of a particular target concept, and the redundant features are the features that provide mostly information that is already given by other features [Yu and Liu 2004]. Furthermore, feature selection can improve the classification performance by eliminating the noise features. Noise features are defined as the features that hurt the predictive accuracy of a particular target concept. In [Zhang and Chen, 2002] experiments, the same or better performance can be obtained after removal of up to 90% of the features in the feature set.

4.2.1 Overview of Feature Selection

Feature selection searches for a subset of features in the feature space to better represent the data. Two main strategies can be used in feature selection are wrappers and filters. The wrapper approach uses an induction algorithm to estimate the value of the attributes, however, this approach is computationally extremely expensive for data with high dimensional feature space [Blum and Langley, 1997]. Therefore, we do not consider this approach in this thesis. On the other hand, the filter approach operates independent of any induction algorithm, and it uses independent scoring criterion to evaluate each

feature [John and Kohavi, 1997]. In this approach, all features are sorted according to their scores, and then the N highest scoring features are selected to form a feature subset, where N is a number predefined by the user. Some common scoring criterion in filtering approach includes Chi^2(Chi-square) testing, IG(Information Gain), (BNS)Bi-Normal Separation, and the Odds-ratio statistic. If computation time is a concern, document frequency is also a good scoring criterion to try, and it can serve as the baseline for comparing feature selection algorithms.

Performing feature selection on the highly skewed data is difficult comparing to the case when the data is balanced. When the class distribution is balanced, most of the feature selection metrics will choose the features proportionately to the class distribution. That means the selected features can represent the documents in each class. However, when the data is highly skewed, the behavior of the feature selection metrics is changed. According to the experimental result in [Tang and Liu, 2005], the IG, Chi^2 and Odds tend to select features that represent the minority class when the data is highly imbalanced, while these methods select features that can represent each class when the data is balanced. Tang and Liu also discovered that the behavior of BNS is not sensitive to the class distribution, and according to their experimental result, BNS outperforms the other metrics when the data is highly skewed. The same conclusion is drawn in [Forman, 2003], when the author performs a study of twelve feature selection metrics on 229 text classification problem instances drawn from 19 datasets that originated from Reuters, OHSUMED, and TREC. Since the above studies do not have emphasis on recall, they both evaluate the performance of the experiment using F-measure. In our study, we are more interested in obtaining high recall at reasonable precision, so we will do the similar comparison of these metrics using our data.

In our experiments, we observed that the proportion of minority class features in the feature set directly affects the performance of the classification. To study how the performance changed by changing the feature ratio, we modified the BNS algorithm to select positive and majority class feature separately. In the following section, we will

discuss the theory of the BNS algorithm and the extended version of this algorithm. The comparison of the performances for these two algorithms will be shown in Chapter 5.

4.2.2 Bi-Normal Separation (BNS)

BNS is a new feature selection metrics introduced by George Forman in 2003 [Forman, 2003]. We denote the number of the positive examples as *pos*, the number of the negative examples as *neg*, the number of positive examples containing a specific word as *tp*, and the number of negative examples containing a specific word as *fp*. The scoring of BNS for this word can be expressed as

$$| F^{-1}(tpr) - F^{-1}(fpr) |$$

where $tpr = tp/pos$ and $fpr = fp/neg$. F^{-1} is the inverse cumulative probability function for a normal random variable. Suppose the occurrence of a feature in each document is modeled by the event of a random normal variable X exceeding a threshold x, for any number p between 0 and 1, F^{-1} returns the threshold x such that some random variable has probability p of being less than or equal to x, $P(X <= x) = p$. Since every cumulative distribution function is monotone increasing and continuous from the right, the larger the p the larger the x. If a feature occurs more frequently in the negative class than in the positive class, the value of $F^{-1}(fpr)$ would be larger than $F^{-1}(tpr)$, and vice versa. The BNS method measures the separation between the thresholds for *tpr* and *fpr*, as shown in Figure 4.2. To avoid the undefined case $F^{-1}(0)$, Forman suggested using 0.0005 to substitute for 0.

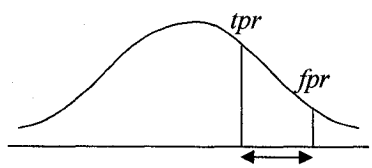


Figure 4.2 Normal Probability distribution

BNS performs well in the class imbalanced situation since it takes class distribution into account. It normalizes a feature's document frequency in the positive and negative class

by the total number of positive and negative examples, so the weight of a feature is also affected by the size of the class. It takes the absolute value of $F^{-1}(tpr)$ and $F^{-1}(fpr)$, so that the larger the difference between these two values, the more likely the feature will be selected. In other words, the selected feature must either have a strong influence in the positive class or in the negative class.

4.2.3 Modified Bi-Normal Separation

As we described above, the selected features in BNS are the ones either strong in the positive class or the negative class. In BNS, a *minority class feature* is a feature that appears more frequently in the positive class, i.e. $F^{-1}(tpr) > F^{-1}(fpr)$, and *majority class feature* is a feature appears more frequently in the negative class, i.e. $F^{-1}(fpr) > F^{-1}(tpr)$. One of the success points of BNS is that the proportion of its selected minority class feature and majority class features tends to conform to the smoothed class distribution. That is because the majority class is more likely to have more strong features. However, when the class distribution is extremely biased, there could be only a few of minority class features selected. For example, a feature set with 100 features could contain only one minority class feature if the class distribution is 1 to 100. This feature ratio is precarious when the minority class is much more important or the negative examples in the training set cannot represent the whole negative class. This leads to a question: how can we get a better result by tuning the ratio between the positive and majority class features?

The motivations to modify the BNS metrics are as follows:

1. Study the performance while changing the feature ratio.
2. The user can have better control of the ratio between the minority class features and majority class features.
3. The BNS cannot change the weight of a feature according to the cost of the minority class.

We modify the BNS by selecting the positive and majority class features separately, and try to apply the smoothing function proposed in [Tang and Liu, 2005] to get the optimal feature ratio. This function smoothes the class distribution when the class is highly imbalanced, and we use this smoothed class distribution as the optimal feature ratio:

$$\text{Optimal feature ratio (OFR)} = \frac{1}{1 + e^{(-\alpha(p-0.5))}}$$

where p is the class percentage, and the α is a parameter to control the degree of smoothing. In their experiment, the optimal value for α is in range 4 to 7. By looking at this function closely, the feature ratio could change radically by changing the smoothing parameter α. To find the behavior of this function, we tried to apply this function to different class distributions from 0.02 to 0.5 by changing the value of α from 1 to 8.

The smoothed class distribution is shown as follow:

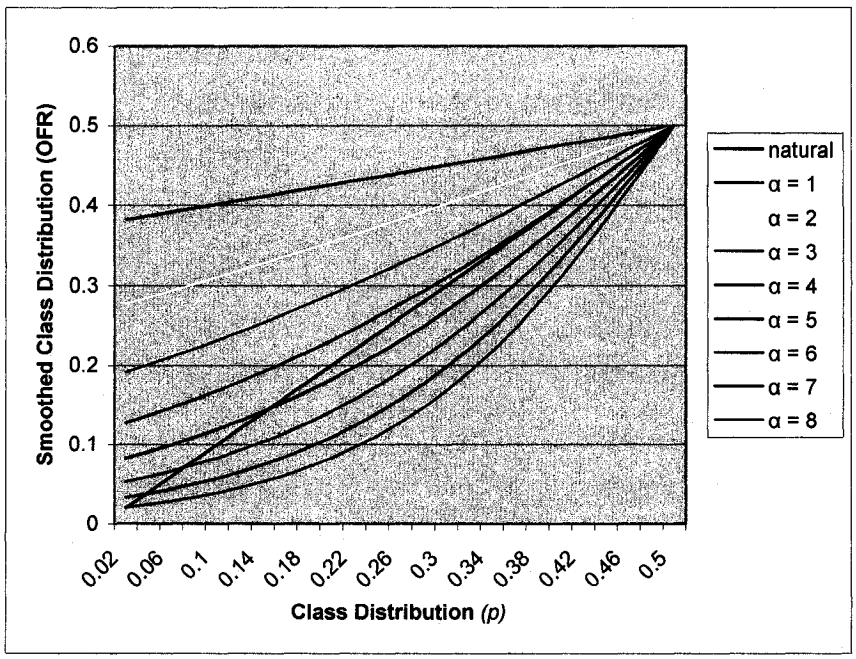


Figure 4.3 smoothed class distribution for various class distributions

From the *Figure 4.3*, we can see that this function is very sensitive to the value of α when the class is highly skewed. As shown in *Table 4.1*, when the class distribution is 0.02, the smoothed class distribution can change from 0.382 to 0.021 by using different α. Therefore, choosing a right value of α becomes the key to find the optimal feature ratio.

Natural Class distribution	α = 1	α = 2	α = 3	α = 4	α = 5	α = 6	α = 7	α = 8
0.02	0.382	0.277	0.192	0.128	0.083	0.053	0.034	0.021

Table 4.1 smoothed class distributions with the class distribution equals to 0.02

The other observation is that the smaller the value of α, the more even the feature ratio we can get. From the *Figure 4.3* we can see that the slope the smoothed class distribution becomes steeper and steeper by increasing the value of α. The user can choose different value of α according to the importance of the minority class. Reducing the value of α can increase the number of minority class features to be selected.

Our intention to modify the BNS is to have more positive examples when the data is extremely imbalanced. However, the smoothing function can do this job well only when the value of α is small. For α greater than 4, the function will choose more majority class features than the natural class distribution at a certain point, and the greater the α, the smaller this point would be. For example, when the α equals to 7, the feature ratio obtained by smooth function chooses fewer minority class features than the natural class distribution for any class distribution greater than 0.038.

In conclusion, the value of α plays an important role in obtaining the optimal feature ratio. We should choose the value of α according to the original class distribution. In the other point of view, the user can change the value of α according to the cost of the minority class. More emphasis will be put on the minority class if a small α is chosen.

In our first dataset, the cost for the minority class is high and the class distribution is 0.05. Our hypothesis is that choosing a small α will give us a better result. To prove this

hypothesis and find the relation between the class distribution and the smoothing parameter α, we will experiment with different feature ratios on our dataset. The detailed experimental result will be discussed in the Chapter 5.

4.3 Sample Selection algorithms

After handling the feature selection bias, in this section we will discuss the algorithm we use to handle the class bias. There are two main tasks in this part. The first one is to address the class distribution by using re-sampling techniques. The second one is to apply active learning to select the most informative examples to be labeled so that the required number of labeled examples is reduced.

4.3.1 Re-sampling

The purpose of re-sampling is to balance class distribution by increasing the minority class's frequency or decreasing the majority class's frequency in the training data. Common re-sampling techniques include random over-sampling, random under-sampling, directed over-sampling and directed under-sampling.

Random over-sampling randomly reproduces the minority class examples to balance the class distribution. The drawback of this method is that it increases the likelihood of overfitting, and it will increase the computation time if the dataset is already very large [Kotsiantis and Kanellopoulos, 2006]. Alternatively, random under-sampling balances class distribution by randomly deleting examples in the majority class. The disadvantage of such method is that it potentially deletes important examples. Nevertheless, under-sampling is still found to be a very useful method to solve the imbalanced problem. In [Alexandersson et al., 2005], the authors compare under-sampling, over-sampling, boosting and bagging when re-balance the class distribution. They show that under-sampling provides the most stable improvement in accuracy. [Drummond and Holte,

2003] also agree that under-sampling outperforms the over-sampling. In this thesis, we apply the idea of under-sampling. In order to overcome the disadvantage of random under-sampling, we use directed under-sampling instead. The details of directed under-sampling will be present in section 4.3.1.2.

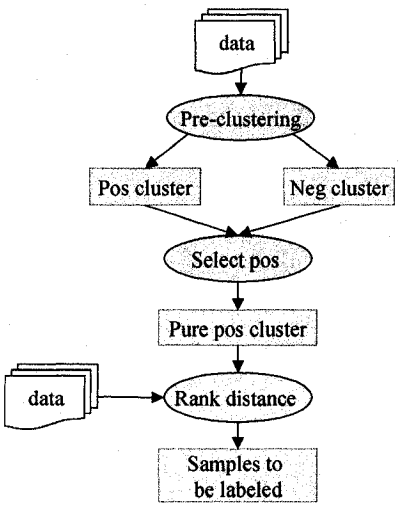
4.3.1.1 Creating initial training set

Prior to under-sampling the data, the initial training set must be formed. As discussed in the introduction, one of the objectives of this thesis is to reduce the number of manually labeled examples to save the cost and time when deploying the system. With highly skewed data, a large number of examples need to be labeled to obtain a certain number of positive examples. In our case, the skew ratio in the data set is around 1 to 20. If we want to have at least 50 positive examples in the training set, around 1000 examples have to be labeled if we pick the examples randomly. We introduce a sample selection algorithm that uses clustering information to solve this problem. This algorithm is referred as *clustering based sample selection* hereafter.

In this algorithm, the training examples are first clustered into two subsets. By the K-means method, an example is assigned to the cluster with the centroid closest to this example. Since the result of the clustering could not be 100% accurate, the center of the cluster may be biased. So the examples in the decision boundary would easy be misclassified, and they are more likely to appear in the boundary of the other class. In other words, the examples around the clustering center are the ones that can nicely represent the cluster and the ones in the cluster boundary are the ones that easily cause confusion. Since we want to have examples that can represent the cluster and we want to have the real label of the difficult examples to reduce confusion, we manually label a small number of examples in the center and the boundary of each cluster. After obtaining a certain number of positive examples, we use them to form a positive cluster. We calculate the distance between this positive cluster and the rest of the unlabeled examples,

and sort them in ascending order. Finally, we select positive examples from the head of the list and select the negative examples from the tail of the list.

The algorithm of forming the initial training data follows:



1. Input: bag-of-words representation of unlabeled documents
2. Pre-clustering: cluster unlabeled data into two data sets using k-means algorithm.
3. Select pos: choose examples in the center and the boundary of these clusters until certain number of positive examples is found. Create a cluster containing only those positive examples.
4. Rank distance: calculate distance (Euclidean / Cosine) between the mean of the pure positive cluster and unlabeled examples. Rank the examples in ascending order. Label predefined number of examples in ascending order.

In the above algorithm, the K-means algorithm is used to form the two clusters. Euclidean distance or Cosine measure is used to calculate the distance between documents. The details of these algorithms are presented below.

K-means algorithm

K-means algorithm is one of the simplest unsupervised learning algorithms that solves the clustering problems. The main idea is to define k centroids, one for each cluster (the centroid is the mean of all the individual records in the cluster); and then assign each record in the dataset to the nearest cluster by measuring the distance to the centroids of each cluster. The next step is to re-calculate the centroids of the clusters resulting from the previous step and re-assign each example in the dataset to the most similar cluster in terms of distance to the centroids. The preceding steps are repeated until the centroids do not move any more. K-means can be performed with Euclidean distance or cosine distance. We choose Euclidean distance as the default similarity measurement due to its efficiency.

The K-means algorithm can be summarized as follow:

1. Place K points into the space represented by the objects that are being clustered. These points represent initial group centroids.
2. Assign each object to the group that has the closest centroid according to the distance.
3. When all objects have been assigned, recalculate the positions of the K centroids.
4. Repeat Steps 2 and 3 until the centroids no longer move.

Figure 4.4 K-means algorithm

Euclidean distance

Euclidean distance is a straight-line distance between two points. If document A and B are represented as point p_1 at (x_1, x_2) and p_2 at (y_1, y_2) respectively, the Euclidean distance between the two documents is

$$d_{AB} = \text{SQRT}((x_1 - y_1)^2 + (x_2 - y_2)^2)$$

Applying the Euclidean distance to documents with more than two dimensions, the above formula can be generalized as

$$d_{AB} = \text{SQRT}(\Sigma(x_i - y_i)^2) \quad \text{where } i = 1, 2 \ldots n$$

If two documents are similar, they will have a small Euclidean distance, and vice versa. Since the distance measurement algorithm is used heavily in clustering algorithms, so the complexity of the similarity measure makes a difference in overall performance. That is the reason we chose Euclidean distance as our default distance measurement algorithm instead of cosine measure.

Cosine measure

Suppose two documents *A* and *B* are represented as two vectors. In Cosine measure, the distance between two documents is captured by the cosine of the angle θ between the two vectors in space.

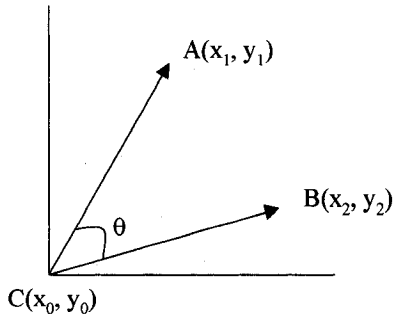


Figure 4.5 Cosine distance between two documents

According to the cosine theorem, cosine θ equals to the dot-product of the vectors normalized by the product of the vector lengths. The cosine similarity between two documents is given by

$$\text{Sim}(A, B) = \text{cosine } \theta = \frac{A \bullet B}{|A||B|} = \frac{x_1 * x_2 + y_1 * y_2}{(x_1^2 + y_1^2)^{1/2}(x_2^2 + y_2^2)^{1/2}}$$

In order to apply the cosine measure to documents with more than two dimensions, the above formula can be generalized as

$$sim(d_j, d_k) = \frac{\sum_{i=1}^{n} w_{i,j} w_{i,k}}{\sqrt{\sum_{i=1}^{n} w_{i,j}^2} \sqrt{\sum_{i=1}^{n} w_{i,k}^2}}$$

When two documents have most words in common, the distance between them is close to one. On the other hand, if two documents do not have any terms in common, the distance between them is zero, which is the opposite of Euclidean distance. In Euclidean distance, the distance between two documents with same words is equal to zero. So if we use

cosine measure, we need to sort the matching documents in descending order of cosine distance.

4.3.1.2 Under-sampling

Once the initial training set is created, under-sampling can be applied to further reduce the size of the subset of negative examples. Since we used the clustering information in the previous stage, the class distribution in the training set is less skewed than the original data. However, under-sampling can further remove the noisy and redundant negative examples in the training set to achieve better performance. There are several directed under-sampling algorithms can be used in our situation.

In [Kubat and Matwin, 1997], the authors divided negative examples into four groups, noisy examples (examples that are misclassified as positive), borderline examples (examples that are on the decision boundary), redundant examples (examples that could be represented by other examples), and safe examples (examples that are worth keeping). The authors suggest that if the first three groups of negative examples were removed, the training set would be safe to apply on classifiers, such as Naïve Bayes or Decision Tree. In their algorithm, they filter redundant examples by removing negative examples that are correctly classified by using 1-Nearest Neighbor. In addition, they filter noisy and borderline examples by removing those examples that participated in Tomek links [Tomek, 1976]. A Tomek link can be defined as follows: given two instances x and y belonging to different classes, and let $d(x, y)$ be the distance between x and y, then a pair $a(x, y)$ is called a Tomek link if there is not a case z, where $d(x, z) < d(x, y)$ or $d(y, z) < d(y, x)$.

In [Zhang & Mani, 2003], the authors suggest four directed under-sampling algorithms besides random under-sampling. The first one is to select negative examples whose average distances to three *closest* positive examples are the *smallest*. The second one is to select negative examples whose average distances to three *farthest* positive examples

are the *smallest*. The third one is to select a given number of negative examples for each positive example. The last one is to select negative examples whose average distances to three *closest* positive examples are the *farthest*. According to their experiments, the second algorithm and the random selection perform the best. However, in terms of recall, the last algorithm has the best outcome.

4.3.2 Active Learning

The purpose of under-sampling is to balance the distribution of the labeled examples. However, we still have a large number of unlabeled examples that can be used to improve the performance. In active classifier, only a small portion of initial labeled examples are needed for building a classifier. At each iteration the active learner will select a few *informative examples* from the unlabeled examples and request for their labels. A human will label these examples and add them back to the training set, and then retrain the learner using the new training set. The iteration will stop when the desired number of training examples is reached. In [Cohn et al., 1996], an *informative example* is defined as an example whose label, when known, can maximally reduce the classification error on unseen examples[4]. In our experiment, these unseen examples are part of the training examples we that set aside.

Active learning is valuable in the situation where obtaining the labeled examples of at least one class is expensive, time consuming or difficult. Examples for such application can be web directory maintenance, e-mail classification, or spam filtering. There are several studies on applying active learning on text classification. One of the approaches is Query-by-committee (QBC), which is introduced by [Seung et al., 1992; Freund et al., 1997]. This approach aims to reduce the expected error in future testing data. In [MacCallum and Nigam, 1998a], QBC is used in combination with EM. According to their experimental result more than half of the training data can be reduced while achieving the same accuracy. However, the authors also claim that if the classes do not

correspond to the natural clusters of the data, using EM can reduce the performance. Another approach is Uncertainty Sampling, which introduced by [Lewis & Gale, 1994]. This approach selects those examples on which the current learner has the lowest certainty. Also, in [Tong and Koller, 2001], the authors perform active learning with Support Vector Machines, which select the sample that halves the permitted region of the SVM parameters in the parameter space. The limitation of this method lies in the SVM version space. The hyperplanes that separate the data in the induced feature space only exist if the training data are linearly separable in feature space. Due to the strong theoretical foundation of QBC, it is selected for our solution. In [Seung et al., 1992], the authors show that using QBC decreases the prediction error exponentially with the number of queries, while [Freund et al., 1997] shows that such exponential decrease is guaranteed for a general class of learning problems. The experimental data for the above studies are relatively balanced. In our study, we test the performance of the QBC on the highly imbalanced data.

4.3.2.1 Query by committee

As we discussed above, active learning reduces the classification errors, and those errors can be decomposed into *bias* and *variance*. In [Geman et al., 1992], the author defined the *bias* of a learner as the loss incurred by the main prediction[5] relative to the optimal prediction. The *variance* of an example is defined as the average loss incurred by predictions relative to the main prediction. Assuming the classifier is not biased, then reducing the classification error is equivalent to reducing the variance over data distribution [Nigam, 2001]. In Query by committee, an example is selected to be labeled when it has the highest classification variance, and the classification variance is measured by the disagreement between a committee of classifiers. Generally, the number of learners in the committee equals to the number of classes. The main idea of Query by

4 This can be determined by training the classifier on the original labeled training set, using this classifier to label the remaining examples, and choosing those which are classified with the least confidence.

Committee is that the information gain of the query is maximized when the disagreement among the committee is maximized.

We denote the labeled training examples as E_l, unlabeled training examples as E_u, the number of classifiers in the committee as n, and the number of examples to be labeled as k. The QBC algorithm is as follows:

Repeat k times:

1. Generate a committee of n classifiers with E_l
2. Classify all examples in E_u with each member in the committee
3. Calculate the disagreement between committee for all examples in E_u
4. Pick the example i with the highest disagreement
5. Ask for its true label for i
6. Remove i from E_u and add it to E_l

Figure 4.6 Query by Committee algorithm

There are three main issues when applying the QBC: (1) how to generate the learning committee and what to use as the base learner, (2) how to measure the disagreement between the committee, and (3) how to select an example to be labeled.

4.3.2.1.1 Generate committee and base learner

In the early study of [Seung et al., 1992] and [Freund et al., 1997], the Gibbs algorithm [Seung et al., 1992] is used as the base learner. Gibbs predicts the label of an example x according to the hypothesis h randomly picked from the version space V. The random selection of h is according to the prior distribution restricted to the version space V.

---

5 The main prediction is the prediction that the learner makes most frequently, and the optimal prediction is the prediction that minimizes the loss taken over all possible values of true classes weighted by their probabilities.

Since h is randomly picked at each time to predict x, so two calls to Gibbs with same V and x can have different predictions. In QBC, Gibbs will be called with different x until disagreement among the committee occurred. There are two main problems by using the Gibbs algorithm. The first is its computational complexity, because the time to search a query diverges with the inverse of the generalization error [Seung et al., 1992]. The second problem is that Gibbs cannot be used with a deterministic component learning algorithm. In [MacCallum & Nigam, 1998a]'s study, Naïve Bayes is used as the base learning algorithm, while the committee of classifiers are generated according to the distribution of classifier parameters specified by the training data. The diversity of the committee is very important to the QBC, since it would be meaningless if we have a committee whose members tend to agree with each other. For this approach, the generation of the committee depends mainly on the training data. Therefore, if the training data does not represent the whole data well, then the distance between the committee members is not maximized. In our study, the data is highly imbalanced and there is no guarantee that the class distribution in the training set would be the same as testing set. Thus, we use another algorithm, Active-Decorate, to generate the learning committee.

4.3.2.1.2 Active-Decorate

Active-Decorate is one of the possible implementations of QBC, and is introduced in [Melville and Mooney, 2004]. The authors use the "Decorate" algorithm to generate the committee. The main idea of Decorate is to use some artificial training data combined with the original training data to train the base learner such that the diversity of the learners in the committee is maximized. In [Zenobi & Cunningham, 2001] the authors show that increasing diversity would decrease ensemble error. The artificial training data is generated according to the distribution of the original training data, and is labeled with the probability of class labels inversely proportional to current committee's prediction. In other words, if the current committee says an artificial example is 80% positive, the new learner will say it is only 20% positive and 80% negative. Since the new learner

disagrees with the current committee for all the artificial examples, we are not sure if adding it into the committee will harm the performance. Therefore, when a new learner is created, it will be added to the committee and the ensemble error will be calculated. If adding the new learner decreases the error, the new learner will be added to the committee permanently.

In [Melville and Mooney, 2004]'s experiment, Decision Tree is used as the base learner of the Active-Decorate. In our system, Naïve Bayes is chosen as the base learner of the Active-Decorate. The reason is we performed feature selection and re-sampling to address the class bias problem before proceeds the active learning stage. In [Tang and Liu, 2005], the authors study the impact of feature selection and re-sampling on these Decision Trees and Naïve Bayes. According to their experimental results, Decision Trees are sensitive to re-sampling but insensitive to feature selection. That is because Decision Trees have their own feature selection mechanism in which a feature that has high information gain will be selected. Naïve Bayes, on the other hand, is sensitive to both feature selection and re-sampling. Re-sampling has significant influence on Naïve Bayes since it changes the global class distribution and prior class probability. By the experimental result in [Mladenic & Grobelnik, 1999], feature selection also has significant influence on Naïve Bayes. In our experiments, we compared the results of using Naïve Bayes, Decision Tree and SVM as the base learner of the Active-Decorate, and we found Naïve Bayes outperforms the other two methods.

The algorithm for the Active-decorate is as follows:

$Committee_{Initial}$ = *BaseLearn*(*T*)

Iterate when i < *NumLearner* and trials < *NumIteration*

1. Generate *Fraction* * |*T*| artificial examples, where *Fraction* based on distribution of T

2. Label *AT* with probability of class labels inversely proportional to current committee's prediction

3. Create new classifier, C, from $T = T \cup AT$

4. Add C to current committee and calculate ensemble error

5. T = *T* - *AT*

6. If error increases

   Remove C from the committee

Figure 4.7 Active-decorate algorithm

where we denote the base learner as *BaseLearn*, the training data as *T*, the artificial training data as *AT*, the desired number of learners in the committee as *NumLearner*, the maximum iteration as *NumIteration*, and the fraction of training data as *Fraction.*

4.3.2.1.3 Disagreement measurement

Disagreement measure part in the QBC is responsible for calculating the disagreement between the members in the committee. The *Margin* is used to measure the disagreement in [Abe & Mamitsuka, 1998]. The *Margin* is the difference between the highest and second highest predicted probabilities from the members in the committee. This criterion is simple and straightforward, but the drawback is that it does not take into account the confidence of the classification produced by committee members. An alternative approach is Kullback-Leibler divergence, which measures natural distance $D(P||Q)$, between a true probability distribution *P* and an arbitrary probability distribution *Q*.

Typically P represents data, observations, or a precisely calculated probability distribution, while Q represents a theory, a model, a description or an approximation of P. In QBC, P represents a posterior class distribution $P_m(C|d_i)$ for a committee member m, and Q represents the class distribution mean over all committee members $P_{avg}(C|d_i)$. In this way, KL-divergence could measure the strength of the certainty of disagreement by calculating differences in the committee members' class distribution. Thus, the disagreement of the committee could be measured by

$$\frac{1}{k}\sum_{m=1}^{k} \mathrm{D}(P_m(C|d_i)||P_{avg}(C|d_i))$$

where k denotes the number of members in the committee, C denotes the class and d_i denotes a document, and the algorithm for $\mathrm{D}(P||Q)$ is

$$\mathrm{D}(P||Q) = \sum_i P(i)\log\frac{P(i)}{Q(i)}$$

4.3.2.1.4 Document selection

The simplest method to select a document for labeling is to choose the document with the highest disagreement value. The disadvantage of this method is that it ignores the class distribution; in other words, a document that has highest disagreement but is far from others could be selected to label. We avoid selecting such documents, because their true label does not help to improve the performance. In [McCallum and Nigam, 1998a], the disagreement is weighted combined with document density. The document density of a document is the average distance between this document and all its neighbors. If a document is close to its neighbors, it means its document density is high, otherwise, its document density is low. Kernel density estimation (also called Parzen Window) is one of the efficient ways to calculate the document density. Let document d be the center of a hyper-sphere, and r be the radius of the hyper-sphere, to calculate the document density of d means to calculate the average distance between d and all other documents in the hyper-sphere.

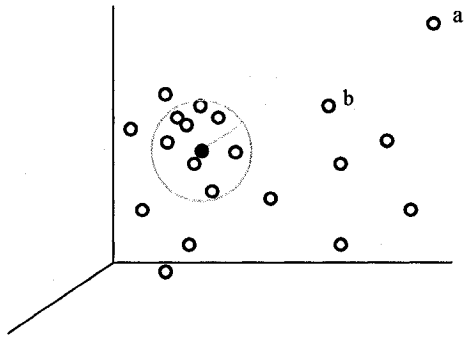


Figure 4.8 Document Density for a given document

As suggested in the *Figure 4.8*, the red dot has higher document density than the blue dot (E.g. blue dot *b*) in the top right corner, so if they have the same disagreement, the red one will be chosen.

Another important observation from the *Figure 4.8* is that the radius of the hyper-sphere is very important. If the radius is too small, the estimation would become less accurate. For example, in the above graph if the radius is small enough, point *a* and point *b* would have the same document density, because the sphere would contain just *a* or just *b*. In fact, the density of point *b* is higher than point *a*. Similarly, if the radius is too large, the estimation would become less sensitive, because the estimated document density would be very close for most of the examples. In [Zhang and Chen, 2002] the optimal r value is suggested as the maximum distance from any object to its closest neighbor, i.e. $r = \max[\min(\|d_i - d_j\|_2)]$. However, this r value is not suitable in our case since we have some documents that share no attribute with others, which makes them very far away from their nearest neighbor. The hyper-sphere with this r almost covers all the documents in the corpus. Therefore, we adjust the r value by dividing it by a constant α. In section 6.3, we experimented with different α to find the optimal radius.

The formula of Kernel density estimation for calculating document density of document d_i can be expressed as follows:

$$DD_i = \sum_{j=1}^{N} \text{Exp}\left[- \frac{\|d_i - d_j\|_2^2}{2r_j^2} \right]$$

4.4 Evaluation measures

4.4.1 Traditional Evaluation Metrics

To evaluate the performance of the proposed system, several traditional evaluation metrics are considered: accuracy, precision, recall, and F-measure. Accuracy is a common measure metric in text classification. It represents the percentage of documents that are correctly classified. However, when the class distribution is highly imbalanced, accuracy cannot measure the performance properly. For example, if the class distribution is 1:9, we can reach a very good accuracy, say 90%, if we simply classify all the documents in the negative class. However, 90% does not represent the quality of the classification, especially when the minority class is important. Precision and recall are alternative evaluation metrics that can tolerate the imbalance. Precision is the ratio of the number of correctly labeled positive examples to the number of all labeled examples. Recall is the ratio of the numbers of correctly labeled positive examples to the number of all positive examples.

In order to present the evaluation metric, we divide the count of classified documents into four groups as follows:

	Classified positive	Classified negative
True positive	TP	FN
True negative	FP	TN

Where:

TP (true positive) is the number of positive documents correctly classified as positive

TN (true negative) is the number of negative documents correctly classified as negative

FP (false positive) is the number of negative documents incorrectly classified as positive

FN (false negative) is the number of positive documents incorrectly classified as negative

The formula for accuracy is

$$\text{Accuracy} = \frac{TP + TN}{TP + FN + FP + TN}$$

Precision and recall are:

$$\text{Precision} = \frac{TP}{TP + FP} \qquad \text{Recall} = \frac{TP}{TP + FN}$$

The target of the classification is to classify the documents into the TP and TN, while minimizing the numbers of documents in FP and FN. In some cases, such as the results of search engines, precision is more important, while in applications such as email classification system, recall is more important. To balance these two evaluation metrics into a single measure of the performance, the F-measure is commonly used. The F-measure combines precision and recall with equal weight as follows:

$$F_0 = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

When the importance of precision and recall is different, the weight of these two elements can be adjusted by α. For example, recall weight is as important as precision when $\alpha = 2$, and precision is twice as important as recall when $\alpha = 0.5$.

$$F_\alpha = \frac{(1 + \alpha) \times \text{Precision} \times \text{Recall}}{\alpha \times \text{Precision} + \text{Recall}}$$

4.4.2 Evaluation Metrics for Systematic Review

The main purpose of applying Systematic Review System is to reducing the human efforts in reading the documents. Therefore, the users would also like to know how much workload they can save after the System is applied. In [Cohen et al., 2006], the authors proposed a new evaluation metrics, work saved over sampling (WSS), from this perspective. Since the recall is very important to the systematic review, so a recall of 95% or higher is assumed to be required in [Cohen et al., 2006]'s study. We will also make this assumption in our studies, hence, a method that is able to achieve a recall of 95% or higher is consider being valuable. The WSS measures the work saved over and above the work saved by simple sampling for a given level of recall.

The WSS is defined as:

$$WSS = (TN + FN)/N - (1 - Recall)$$

where N is the total number of samples in the test set.

4.5 Conclusion

In this chapter, we introduce each component of our text classification system. In Chapter 5, we will try different methods to address the bias in different classification stages. We will study how the bias affect the performance, and compare the performance of different methods. To study how much each method contributes to improve the performance, we will isolate each technique in the experiment. For example, when we test the influence of different feature selections to the performance, the sample selection stage will not be involved in this set of experiments. In Chapter 6, we will include the active learning in sample selection stage and study if the idea of active learning is suitable for imbalanced text classification.

Chapter 5

Impact of bias on text classification

5.1 Introduction

As discussed in the previous chapter, the imbalanced data could lead to bias problems in feature selection, sample selection and classification. To address these issues, we first studied the feature selection and sample selection bias caused by the skewed data. We then experimented with different feature selection, sample selection, and classification methods to find the ones that can properly handle these problems. We also studied how well these methods can work together to further improve the performance.

We compared the performance of BNS with three other popular feature selection methods: Information Gain, Chi-square, and Odds ratio. We focused our study on BNS because it was proved to outperform the other feature selection methods when the data is highly imbalanced in [Forman, 2003] and [Tang and Liu, 2005]. By the definition of the BNS, it handles the feature selection bias. For the sampling selection bias, we used under-sampling to adjust the bias in the training data. The reasons for choosing under-sampling instead of over-sampling are discussed in section 4.3.1. We experimented with different under-sampling methods and studied how well they work with different feature selections. To adjust the classification bias, the Complement Naïve Bayes is applied. This classifier was also tested in combination with different feature selection methods.

5.2 Methodology

The performance measures such as precision, recall and accuracy for the minority class are used in each experiment.

For each experiment, we randomly selected three distinct subsets of documents from the original dataset for selecting features, training classifiers and testing respectively. Each subset contains 2500 documents. We do not combine the training data for feature selection and classifier because the training data's label for classifier is assumed to be unknown when performing the active learning in Chapter 6. To better compare the results of active learning with the approaches in this chapter, we use the same way of obtaining the data in Chapter 5 and Chapter 6. The same experiment is performed five times with different data subsets, and the average of these performances is taken as the final result. A document is represented as a vector of binary values, where each feature indicates whether a given word in the vocabulary is presented in the document or not. The frequency based representation is not used in here because it is more suitable to the case when the number of attributes is large [McCallum and Nigam, 1998b]. In our studies, the number of attributes is reduced by features selection to a relatively small pool, so binary representation is chosen.

We used the implementation of the induction algorithms such as Naïve Bayes, Decision Trees and Support Vector Machines in Weka 3.5. The default configurations of these algorithms are applied. The implementations of Chi-square and Information Gain with default setting in Weka are used in this experiment. Since Odds Ratio and Bi-normal Separation are not implemented in Weka 3.5, we implemented these methods according to the algorithms described in [Forman, 2003].

5.3 Experiment results baseline

The baseline of our experiment is the precision, recall and f-measure of the minority class when classifying all documents as the minority class. In our data, only 5% of the documents are in the minority class, so the precision in the baseline is 5%. Since all documents are classified as the minority class, the recall is 100%. The f-measure equals (2 * precision * recall) / (precision + recall), which is 9.5%.

	Precision	Recall	F-measure	Accuracy
Baseline	5%	100%	9.5%	5%

5.4 Adjusting feature selection bias

In the first part of this section, we studied the feature selection bias for some popular feature selection methods in the imbalanced data situation. The feature selection bias occurs when most of the selected features represent only one of the classes. In the biased feature set, the strong features in one class hide the useful features from the other class. In the second part, we experimentally investigated the impact of feature selection bias on the classification performance. Hereafter, we refer feature selection methods Chi-square, Information Gain, Odds-ratio and Bi-normal Separation as Chi, IG, Odds, and BNS respectively.

In order to reduce feature space and save computation time, all stop words[6] are removed before applying feature selection methods in all the experiments. Stop words are the non-informative words that appear very often in the document, such as prepositions, conjunctions, auxiliary verbs etc. Stop words are not useful in text classification because they can appear in any class and are irrelevant to the content of the documents.

6 Used stop words in MSDN library. http://msdn2.microsoft.com/en-us/library/bb164590(VS.80).aspx

5.4.1 Feature selection bias

To better study the feature selection bias, we divided features into two exclusive classes, minority class features and majority class features. Minority class features are the features that occur more frequent in the minority class than in the majority class, while majority class features are the features that occur more frequent in the majority class than in the minority class. The ratio between minority class features and majority class features in the feature set is referred as *feature ratio* hereafter. To study the *feature ratio* of Chi^2, IG, Odds and BNS, we ran our experiment over the same datasets using these feature selection methods.

In imbalanced situation, if absolute term frequency is used as the criterion to select features, majority class features have higher probability to be selected. However, it is not the case when feature selection methods are applied. In [Tang and Liu, 2005], the authors studied feature ratios of Chi^2, IG, Odds, and BNS in data "cora36" with data skewness ratio 1:35. According to their experiment results, Chi^2, IG, and Odds tended to select large proportion of minority class features when the total number of selected features is small. We perform similar studies on our "Nutrition" data, and explore the reason that lead to the bias. In section 5.1.2, we further studied how the feature selection bias affects the precision and recall.

Obtaining minority class feature and majority class feature

We ran our experiment on "Nutrition" dataset, which contains 14,276 documents with data skewness ratio 1:20. To study the feature ratio, we first categorized all features into two groups: minority class features and majority class features, where minority feature is the feature appears more frequently in the minority class examples and majority feature is the feature appears more frequently in the majority class examples. We refer to the number of minority examples that containing a given feature as *tp*, and to the number of majority examples that containing a given feature as *fp*. In our experiment, a document is represented by a binary vector that indicates whether a word occurs in the document. Therefore, a word will be only counted once in *tp* or *fp* even it occurs more than one time

in the document. Let us denote the number of minority examples as *pos*, and the number of majority examples as *neg*. A feature is classified as minority class feature if *tp/pos* > *fp/neg*, and a feature is classified as majority class feature if *tp/pos* < *fp/neg*. We ignore those features where *tp/pos* = *fp/neg*, because they do not represent any class.

We randomly selected a subset of 2500 documents from the original data set and applied the feature selection methods Chi^2, IG, Odds, and BNS on it respectively to obtain four different feature sets. To find the feature ratio obtain by these feature selection methods in different number of selected features, we selected the number of features from 10 to 1000 with increment of 10. We ran this experiment five times to take the average feature ratios and the absolute number of selected minority class features. The results are reported in Figure 5.1 and Figure 5.2.

In Figure 5.1, the x-axis is the number of features, and the y-axis is the percentage of minority class features in the selected feature set.

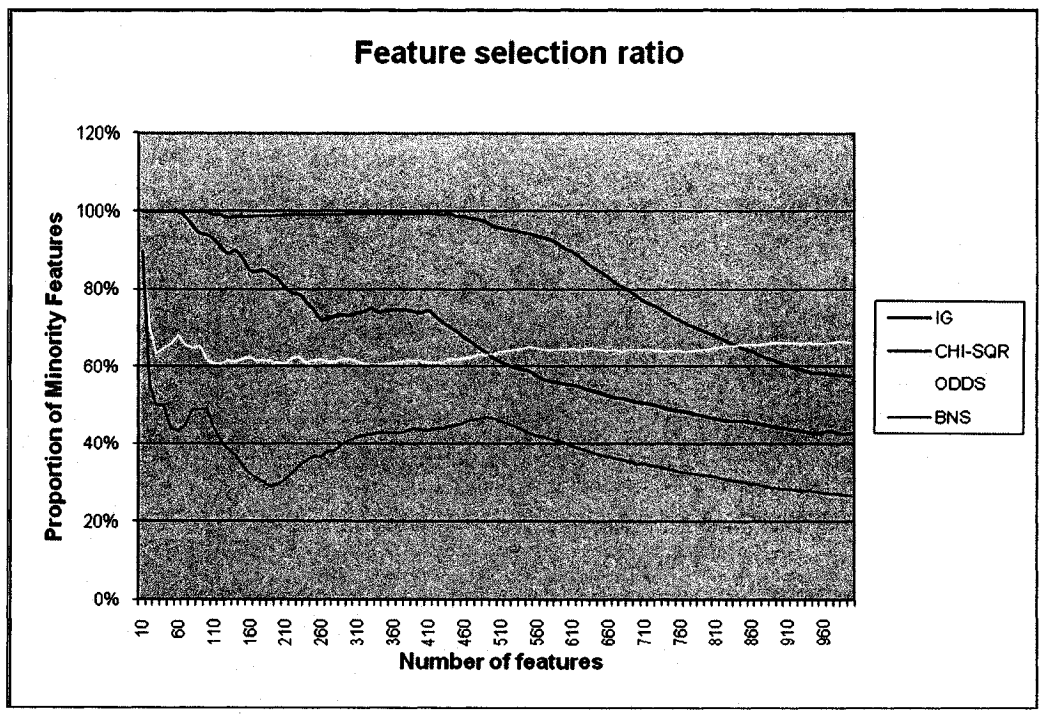


Figure 5.1 Percentage of minority feature in different feature selection methods

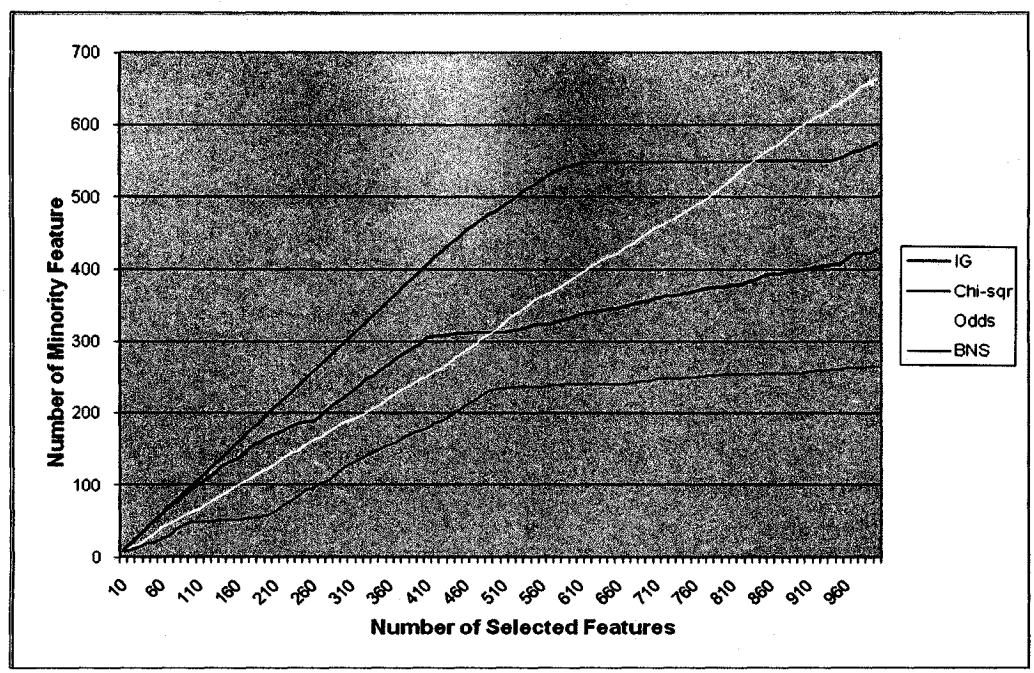


Figure 5.2 Absolute number of minority feature in different feature selection methods

In Figure 5.2, the x-axis is the number of features, and the y-axis is absolute number of selected minority class features in the feature set. The result of Figure 5.2 will be used in section 5.3.2 when study the impact of the feature selection bias on recall and precision.

From the Figure 5.1 we observed that most of the features selection methods tend to select minority class features when the number of features is small. For example, all the features chosen by IG are minority class features when feature set is smaller than 100, and the proportions of majority class features only increase slightly when the number of features increases. Chi^2 tends to have the same behavior, but with higher increasing rate of the proportions of majority class features when the size of the feature set increase. The behavior of odds-ratio and BNS are similar, they both stay in a steady feature ratio, and do not change while the number of feature increases. The odds-ratio chooses around 60% of the minority class feature, and BNS chooses around 40% of the minority class features. The BNS has a reasonably balanced features ratio compared to the others.

Why two popular feature selection methods Chi^2 and IG tend to select minority class features when the feature set is small in the data imbalance situation? Among all the

terms in the entire corpus, let P(t, c) denote the probability of term t and class label c occurring at the same time, P(t, $\bar{c}$) denote the probability that t occurs without c, P($\bar{t}$, c) denote the probability that c occurs without t, and P($\bar{t}$, $\bar{c}$) denote the probability that neither t or c occurs. The rating criteria of Chi^2 and IG mainly depend on the above four probability values.

Take the formula of Chi^2 as an example:

$$\chi^2(t, c) = \frac{N * (P(t, c) * P(\bar{t}, \bar{c}) - P(\bar{t}, c) * P(t, \bar{c}))^2}{P(t) * P(\bar{t}) * P(c) * P(\bar{c})}$$

To achieve a high Chi^2 rating, a feature must have high value of P(t, c) and P($\bar{t}$, $\bar{c}$), but low value of P($\bar{t}$, c) and P(t, $\bar{c}$), which means this feature is strong in one class but weak in other class. Let us look at the case for minority class feature and majority class feature separately.

Case 1: **t** is a **minority class feature** and **c** is the **minority** class.

First, let us look at the numerator in the formula. In *Case 1*, the value of P(t, c) is small but P($\bar{t}$, $\bar{c}$) is large. The value of P($\bar{t}$, c) is small because the number of minority class documents is small. Also, the value for P(t, $\bar{c}$) is small because minority class features occur more frequent in the minority class than in the majority class so that the probability of both minority class feature and majority class label is small.

For the denominator in the formula, P(c) and P($\bar{c}$) would be the same for both *Case 1* and *Case 2*. By the definition of the minority class feature, the probability P(t) for a minority class feature t would be small and P($\bar{t}$) would be large.

Case 2: **t** is a **majority class feature** and **c** is the **majority** class.

In the numerator, the value of P(t, c) is large, but hardly as high as the value of P($\bar{t}$, $\bar{c}$) in *Case 1* due to the diversity and large amount of features in the majority class. The value for P($\bar{t}$, $\bar{c}$) and P(t, $\bar{c}$) are small because the number of documents in the minority class is small. However, the value of P($\bar{t}$, c) is could be large because there are lots of other majority class features exist.

In the denominator, the probability P(t) for a majority class feature t more likely be larger than the probability for the minority class feature. The value of P($\bar{t}$) would be relatively smaller, because the sum of P(t) and P($\bar{t}$) is the same for all the features t.

To summarize the above two cases, the feature that occurs frequently in one class, but seldom occurs in the other class has a high Chi^2 rating, because it maximizes the value of P(t, c) and P($\bar{t}$, $\bar{c}$). When the data is highly imbalanced, those strong minority class features that occur in most of the minority class but seldom occur in the majority class would have a highest Chi^2 rating. As described in *Case 2*, the strong majority class features are unlikely to have the highest Chi^2 rating due to the diversity and large amount of features in the majority class. However, by increasing the number of selected features, those strong majority class features will be selected because the number of strong minority class features is limited.

5.4.2 Impact of the feature selection bias on recall and precision

As discussed above, most of the popular feature selection methods tend to select large proportion of minority class features. In the following section, I will study how this feature selection bias affect the performance of the classification on the minority class. We will analyze how different feature ratio or different number of selected features affects the precision and recall. We will use the feature sets obtained by Chi^2, IG, Odds, or BNS in the above experiments to represent the data for classification. In corpus "Nutrition", we randomly and disjointly generate 2500 documents as training set and 2500 documents as testing set. In this experiment, Naïve Bayes with default setting is used as the classifier. The recall, precision and F-measure of the minority class for different feature selection methods with different number of selected features are shown in the following Figures.

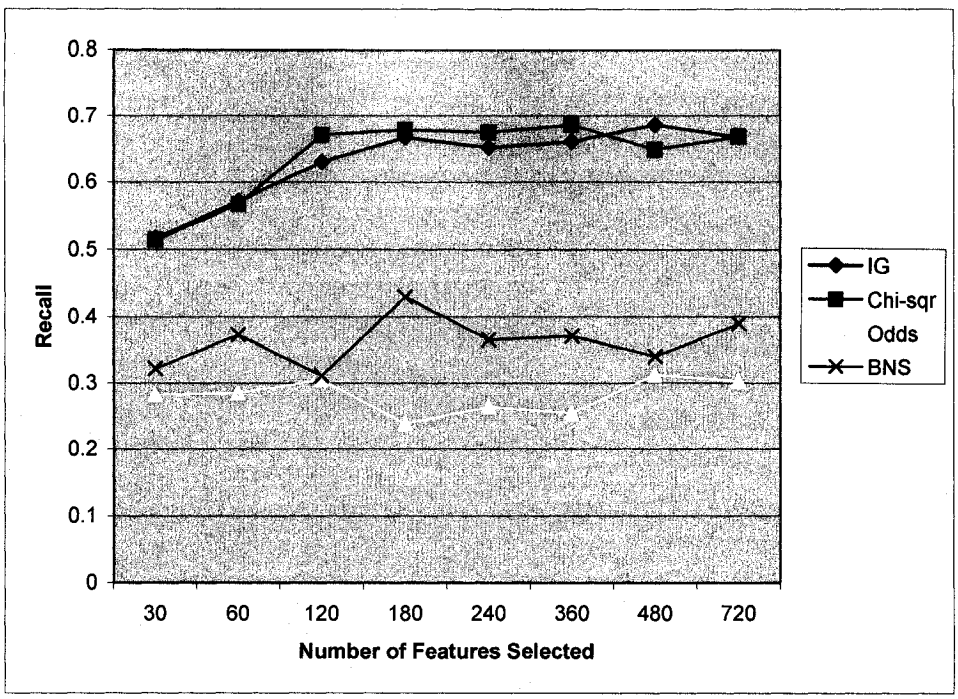


Figure 5.3 Recall of the minority class for different feature selection methods with Naïve Bayes

In terms of recall for the minority class, IG and Chi^2 are significantly outperforming the other feature selection methods as shown in Figure 5.3. Referring to the feature ratios in Figure 5.1, IG and Chi^2 select more minority class features into feature set comparing to Odds and BNS. With a feature set mainly representing the minority class, the classifier would tend to classify a document as the minority class. However, increasing the number of minority class features in the feature set is only one of the factors that improve the minority class recall. It is unnecessarily true that the more minority class features in the feature set, the higher the recall we can get. For example, we cannot get 100% recall by simply classifying on pure minority class features. Refer to Figure 5.2, the number of minority class features for IG and Chi^2 increase almost linearly with the number of selected features. However, their recall stops increasing when the number of selected features exceeds 120. Refer to Figure 5.1, when the number of selected features is 120, the proportion of minority class features is close to 100%, which implies most of these 120 features are minority class features. That means a limited subset of minority class features makes most of the contribution to obtain the high recall.

The other observation from Figure 5.3 is that the number of selected features does not have too much influence on the recall, especially for the Odds and BNS. For the IG and Chi^2, the recall is not affected by the number of selected features after the number of selected features is greater than 120.

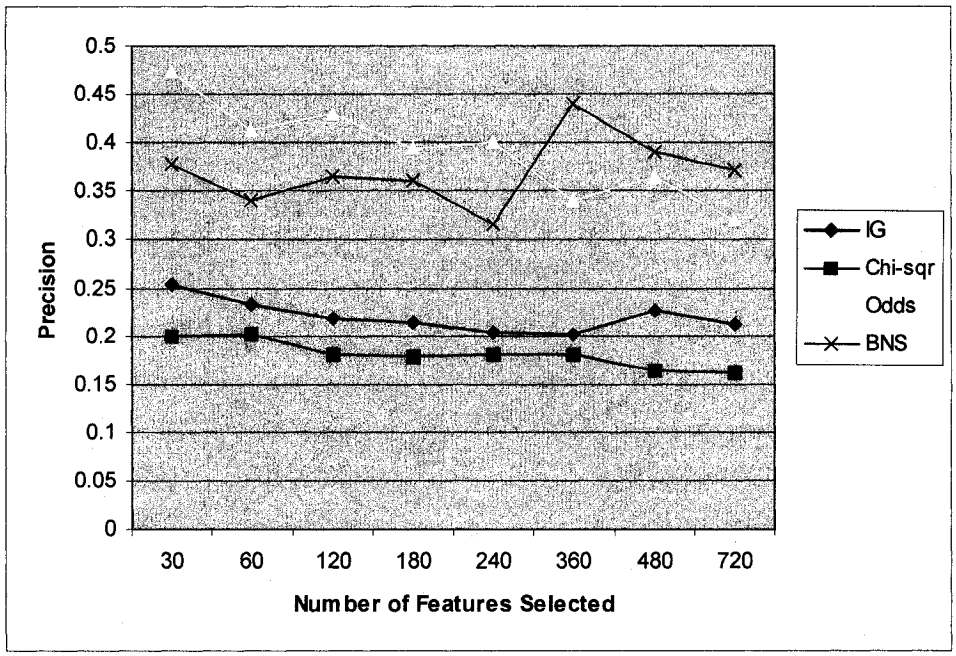


Figure 5.4 Precision of the minority class for different feature selection methods with Naïve Bayes

According to Figure 5.4, Odds and BNS significantly outperform IG and Chi^2 in terms of precision. Since IG and Chi^2 select large proportion of minority class features to represent the data, the classifier tends to classify documents as the minority class, which make the precision very low. On the other hand, Odds and BNS select feature evenly, so that there are enough features to represent each class. Chi2 has the lowest precision, but it achieves the highest recall as shown in Figure 5.3.

The solution with very high recall but very low precision or vice versa is not ideal. We want to find a solution that can balance the tradeoff between recall and precision. F-measure is a measurement criterion that combines precision and recall. According to Figure 5.5, BNS outperforms the other feature selection methods in terms of F-measure

when the number of features is greater than 180. The outstanding performance of BNS in data imbalanced situation is also reported in [Forman, 2003] and [Tang & Liu 2004].

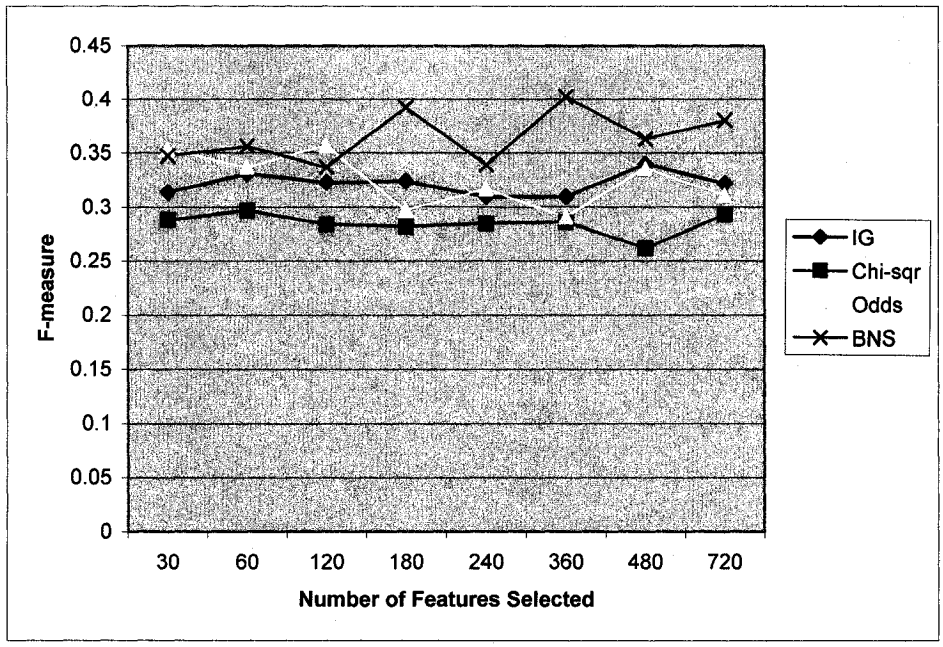


Figure 5.5 F-measure of the minority class for different feature selection methods with Naïve Bayes

In conclusion, the common feature selection methods select large proportion of minority class features when the data is imbalanced. This feature selection bias makes classifiers to tend to classify document as the minority class, which improves the recall but drops the precision. Since IG selects the largest proportion of minority class features, it outperforms the other feature selection methods in terms of recall. While BNS select the features from both classes evenly, it has higher precision and F-measure comparing to others.

5.4.3 Modified BNS

According to the above studies, the proportion of minority class features in feature set affects the classification performance. In order to further study how the feature ratio affects the performance of Naïve Bayes, we modified BNS by selecting minority class features and majority class features separately. Since the class bias in our data set is 1 to 20, we tested the different feature ratio from 0.05 to 0.5. In order to study the impact of the number of minority class feature on the recall of the minority class, we tested the positive and majority class feature ratio from 0.95 to 0.5. We also tried the extreme case where only the minority class features are selected. Naïve Bayes with default setting is used as the classifier. In the previous experiment, BNS reached highest F-measure when the number of features is 360. We used 360 as the total number of features, and we experimented with different ratio of minority class features among the 360 features[7].

#Pos: #Neg	Precision	Recall	F-measure
30:330	0.449	0.231	0.305
60:300	0.413	0.238	0.302
90:270	0.427	0.327	0.37
120:240	0.39	0.392	0.391
150:210	0.403	0.447	0.424
180:180	0.362	0.444	0.399
210:150	0.366	0.406	0.385
240:120	0.378	0.404	0.39
270:90	0.348	0.378	0.363
300:60	0.411	0.346	0.376
330:30	0.408	0.304	0.348
360: 0	0.445	0.304	0.361

Table 5.1 Results for modified BNS with different feature ratios

In this experiment, we observed that it is not true that the higher the proportion of minority class features in the feature set, the higher the recall in the minority class can be achieved. According to table 5.1, the best recall and F-measure occurs when the feature ratio is 150 to 210. As discussed in section 4.2.3, this optimal feature ratio can be estimated by using the smoothed class distribution function proposed in [Tang and Liu,

2005] based on the class distribution and the cost of the minority class. In our study the cost for missing the minority class is very high. This is because misclassifying a relevant document as irrelevant document will cause the medical researcher miss valuable research evidence. According to our study of this formula in section 4.2.3, we include more minority class features when the value of α is small. To put more focus on the minority class, we chose the value of α as 1.

$$\text{Optimal feature ratio (OFR)} = \frac{1}{1 + e^{(-\alpha(p-0.5))}}$$

In this formula, p is the class distribution, and α is the smoothing parameter. If we choose α equals to 1 and p equals to 0.05, the estimated optimal feature ratio equals to 0.39. The best feature ratio we obtained from the experiment is 150 to 210, which equals to 0.41. The two feature ratios are very similar.

5.5 Adjusting sample selection bias

In the data imbalance situation, the training set would be biased if we choose training data randomly from the original corpus. For example, if the class distribution is 1 to 20, we could have only 125 minority examples in a training set with 2500 examples. By using the imbalanced training set, the classifier's decision is biased to the majority class, which causes the minority class to have low recall. The results in Section 5.3 for the Naïve Bayes with 360 features, which are obtained from different feature selection methods, are as follows:

	Precision	Recall	F-measure
BNS	0.439	0.371	0.402
Chi-sqr	0.181	0.686	0.286
IG	0.202	0.661	0.31
Odds	0.341	0.254	0.291

Table 5.2 Results for different feature selection methods without under-sampling

7 According to Figure 5.5, BNS achieved highest F-measure when the number of selected features is 360.

In the experiments in this section, we combined the feature selection methods in the previous section with different under-sampling algorithms to adjust the bias in the training data. Our hypothesis is that by adjusting the class distribution of the training set, the recall of the minority class would be improved. Reported in [Tang and Liu, 2005], directly combining feature selection and random over-sampling does not necessary improve the performance. However, in their study, precision and F-measure are the criteria to evaluate the performance. In this thesis, we are more interesting in achieving high recall with reasonable precision. In the following experiments we studied the result of combining different feature selection methods with various under-sampling methods. According the experiment result in the previous section, the number of features does not significantly affect the performance. We chose 360 features in all experiments in this section. We kept on using the Naïve Bayes with the default setting in Weka as the classifier.

We experimented with four feature selection methods in the Section 5.3. Combined with the five under-sampling methods studied in this section, there are 20 different ways to deal with the training sample bias. When performing under-sampling, all the positive examples in the original training set would be kept, and selecting the same number of negative examples in the following ways:

Method 1: Select negative examples whose average distances to three *farthest* positive examples are the *smallest*. (refer as *FS avg distance*)

Method 2: Select negative examples whose average distances to three *closest* positive examples are the *smallest*. (refer as *CS avg distance*)

Method 3: Select negative examples whose average distances to three *closest* positive examples are the *largest*. (refer as *CL avg distance*)

Method 4: Removing those examples that participated in Tomek links (Refer to Section 4.3.1.2). (refer as *Tomek links*)

Method 5: Select negative examples randomly. (refer as *Random*)

The method 4 and 5 are straightforward. To better explain the methods 1 to 3, we described them as pseudocodes as follows:

Method 1:

1. For each negative example *n*
    1.1 Compute the distances from *n* to all positive examples
    1.2 Choose the positive examples p_1, p_2, p_3 such that they are farthest from *n*
    1.3 Compute average distance $nf(p_i, n)$
2. Let $F = \{nf\}$, and ascending sort *F*
3. Choose *k* negative examples in *F* with smallest *nf*.

Method 2:

1. For each negative example *n*
    1.4 Compute the distances from *n* to all positive examples
    1.5 Choose the positive examples p_1, p_2, p_3 such that they are closest from *n*
    1.6 Compute average distance $nc(p_i, n)$
2. Let $F = \{nc\}$, and ascending sort *F*
3. Choose *k* negative examples in *F* with smallest *nc*.

Method 3:

1. For each negative example *n*
    1.7 Compute the distances from *n* to all positive examples
    1.8 Choose the positive examples p_1, p_2, p_3 such that they are closest from *n*
    1.9 Compute average distance $nc(p_i, n)$
2. Let $F = \{nc\}$, and ascending sort *F*
3. Choose *k* negative examples in *F* with largest *nc*.

The result of combining different feature selection methods with different under-sampling techniques are as follows:

Method 1:

	Precision	Recall	F-measure
BNS	0.051	0.94	0.097
Chi^2	0.048	0.835	0.09
IG	0.059	0.92	0.111
Odds	0.082	0.912	0.15

Table 5.3 Results for different feature selection methods with under-sampling method "FS avg distance"

Method 2:

	Precision	Recall	F-measure
BNS	0.036	1	0.069
Chi^2	0.032	0.553	0.061
IG	0.036	0.742	0.068
Odds	0.026	0.825	0.051

Table 5.4 Results for different feature selection methods with under-sampling method "CS avg distance"

Method 3:

	Precision	Recall	F-measure
BNS	0.044	0.957	0.085
Chi-sqr	0.086	0.974	0.158
IG	0.036	1	0.069
Odds	0.055	0.965	0.105

Table 5.5 Results for different feature selection methods with under-sampling method "CL avg distance"

Method 4:

	Precision	Recall	F-measure
BNS	0.121	0.778	0.21
Chi^2	0.103	0.781	0.182
IG	0.069	0.622	0.124
Odds	0.106	0.462	0.173

Table 5.6 Results for different feature selection methods with under-sampling method "Tomek links"

Method 5:

	Precision	Recall	F-measure
BNS	0.049	0.931	0.093
Chi^2	0.105	0.64	0.181
IG	0.089	0.625	0.156
Odds	0.105	0.447	0.17

Table 5.7 Results for different feature selection methods with under-sampling method "Random"

Overall, under-sampling the majority class improved the recall of the minority class, but at the same time dropped the precision of the minority class. Before under-sampling, most of the examples in the training set are majority examples. Since Naïve Bayes's prediction depends on the prior class probability, so before under-sampling its prediction is biased to the majority class. This leads to a low recall for the minority class. However, under-sampling changed the class distribution in the training set, so the prior class probability also changed. This is the reason for recall improving significantly after under-sampling. On the other hand, the negative examples are very diversified in the imbalanced data situation. When we have equal number of positive examples and negative examples in the training set, we might not have enough negative examples to train the classifier. So the classifier labels most of the testing examples as positive examples, which leads a very low precision for the minority class.

In terms of recall, BNS works well with different under-sampling methods. It has highest recall when it combines with the Method 2, but the precision is very low. When it combines with the Method 4, it has a highest F-measure but with a relative low recall. The results of combining BNS with the rest of the methods are very similar.

The Odds Ratio works well with the Method 1 and 3 in terms of recall. When it combines with other under-sampling methods, either did not improve the recall too much, or have a higher recall but very low precision.

For Chi^2 and IG, they have high recall with reasonable precision when they combine with the Method 1 and 3. They do not work well with other under-sampling methods.

In general, the first and third under-sampling methods work well with all feature selection methods. We have a very high recall after performing under-sampling technique. However, we have a big trade off in precision. This means we cannot save too much human effort since the reviewers still have to go through most of the articles.

5.6 Adjusting classification bias

In the previous experiment, Naïve Bayes is used as the classifier. In this section, we used the Complement Naïve Bayes, which adjusted the decision boundary weight to solve the data bias problem. Since the Complement Naïve Bayes handled the imbalanced data problem implicitly, we do not apply the under-sampling technique to adjust the training set's class distribution beforehand.

In this section, we performed similar experiments as Section 5.3. The only difference here is Complement Naïve Bayes is used as the classifier. The recall, precision and F-measure for combining different feature selection methods and Complement Naïve Bayes are as follows:

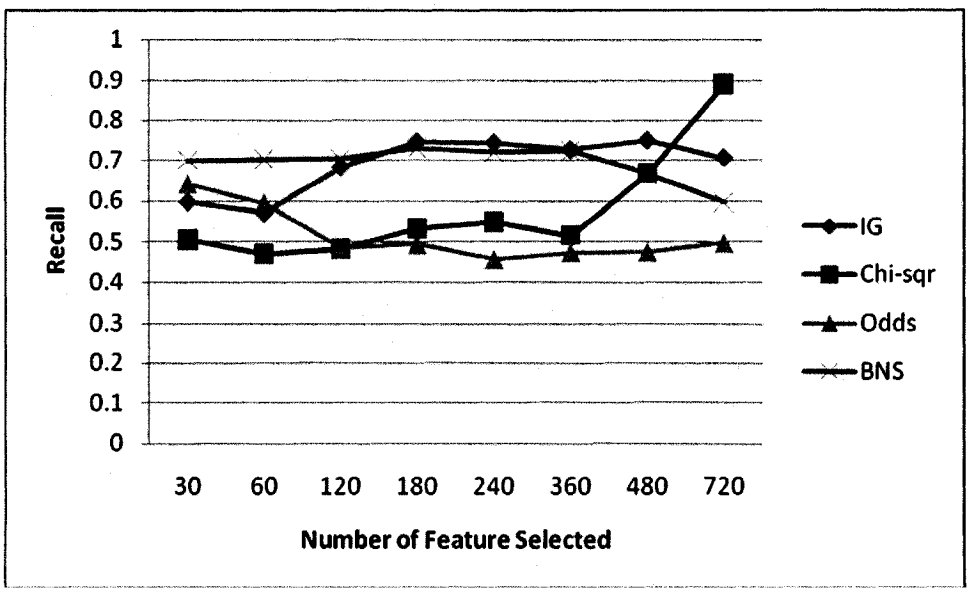


Figure 5.6 Recall of the minority class for different feature selection methods with Complement Naïve Bayes

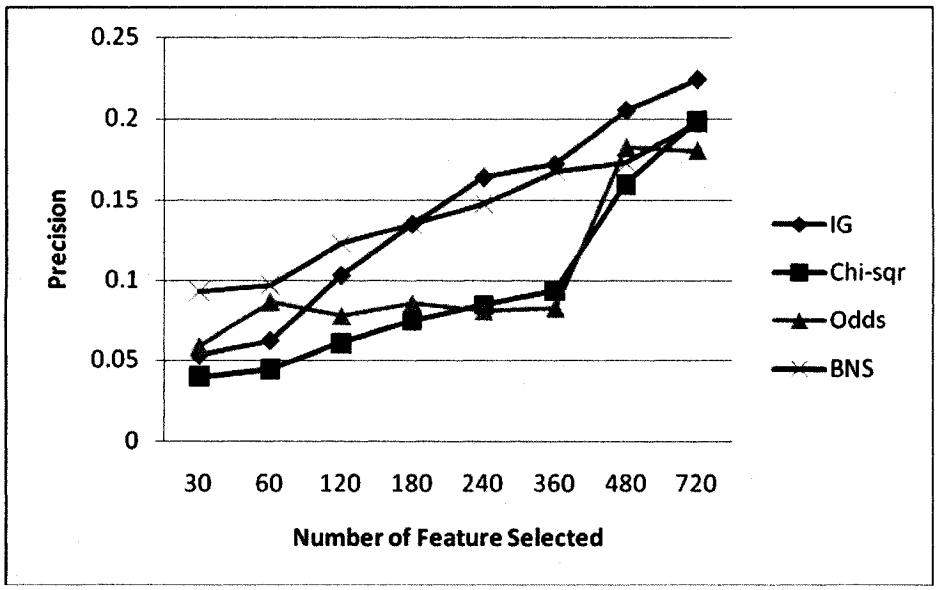


Figure 5.7 Precision of the minority class for different feature selection methods with Complement Naïve Bayes

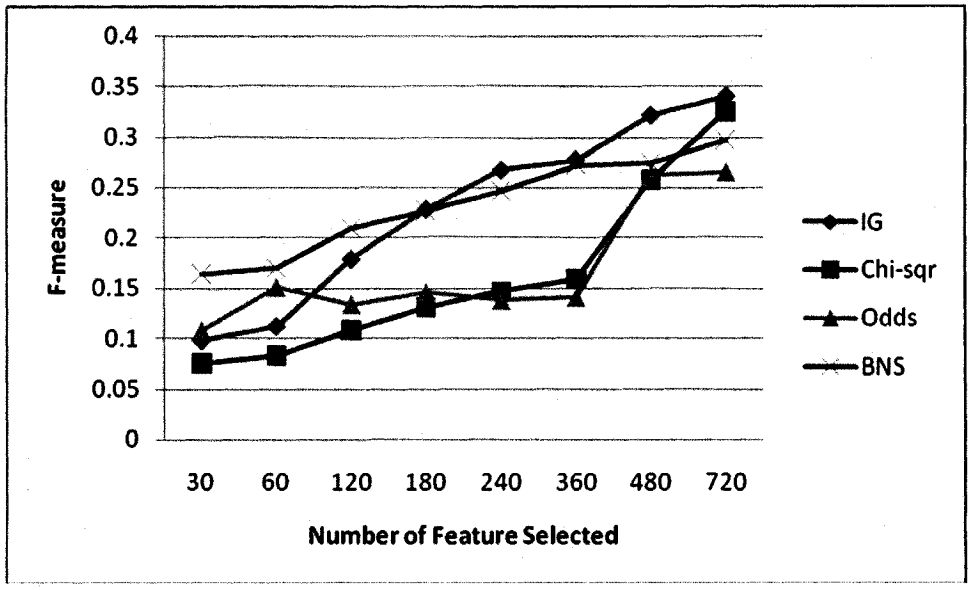


Figure 5.8 F-measure of the minority class for different feature selection methods with Complement Naïve Bayes

Generally, combing different feature selection with Complement Naïve Bayes improved the recall. According to the Figure 5.6, IG and BNS have higher recall when the number of selected features is small, but when the number of selected features is greater than 500, Chi^2 outperforms the other feature selection methods. Odds ratio has the lowest recall among these four features selections, but compared to the recall obtained when it is combined with Complement Naïve Bayes, there's 20% improvement on average. Another interesting finding in Figure 5.7 and Figure 5.8 is the precision and F-measure of the minority class increase linearly by increasing the number of selected features. BNS has higher precision and F-measure when the number of features is less than 180, but when the number of selected feature increases IG has better precision and F-measure.

We also tried to combine the Complement Naïve Bayes and the Modified BNS to study how the feature ratio affects the performance of this classifier.

#Pos: #Neg	Precision	Recall	F-measure
30:330	0.163	0.905	0.276
60:300	0.167	0.906	0.282
90:270	0.149	0.922	0.257
120:240	0.19	0.867	0.311
150:210	0.21	0.895	0.34
180:180	0.187	0.825	0.306
210:150	0.202	0.821	0.324
240:120	0.185	0.684	0.291
270:90	0.179	0.712	0.286
300:60	0.16	0.663	0.258
330:30	0.147	0.565	0.233
360:0	0.088	0.497	0.15

Table 5.8 Results for Modified BNS with Complement Naïve Bayes

According to the result in Table 5.7, the recall decreases when the number of majority class features increases. That's because of the nature of the Complement Naïve Bayes, which is using the data in the negative class to evaluate the likelihood of the positive class. In terms of precision and F-measure, it reaches highest value when the feature ratio is 150 to 210.

Chapter 6

Active learning for text classification

6.1 Introduction

The two main purposes to introduce active learning in this chapter are reducing the error rate and reducing the size of the training set. We have managed to boost the recall of the minority class to 100% by combining feature selection and under-sampling. However, the precision is too low so that the reviewers still have to manually classify many literatures. The second problem is in the previous experiments we used 2500 labeled documents to obtain the feature subset, and used another 2500 labeled documents to train the classifiers. This makes a large overhead to deploy the system, because reviewers have to manually label at least 5000 documents. Since there is exists a high dimensional feature space for text classification, the feature set must be reduced before the rest of the process proceeds. So the 2500 labeled documents for performing the feature selection are a must. However, the second set of the labeled document for training classifier can be reduced by applying the active learning technique. Reported by [Nigam and McCallum, 1998a] and [Melville and Mooney, 2004], active learning can successfully reduce the size of the training set and the error rate. However, in their studies, the active learning system is applied to relatively balanced dataset. In this thesis, we experimented with the active learning technique on imbalanced data.

We used the BNS feature selection method to obtain a feature subset that containing 360 features. Naïve Bayes with default setting in Weka is used as the active learner. Naïve Bayes, Decision Tree and Support Vector Machines with default setting in Weka are applied separately as the classifier. As in Chapter 5, we randomly generate three

exclusive sets of documents for feature selection, training and testing. The difference here is that we assume the 2500 documents for training are unlabeled.

6.2 Obtain training set for active learners.

We used the Query-by-Committee approach to perform the active learning in this thesis. The members in the active learning committee are individual classifiers, and we called them active learners. Before we run the active learning system to choose the next informative example to be labeled, we should have a small training set to train these active learners. If we randomly choose 100 documents to be labeled, according to the class distribution in our data this training set may contain only 5 positive documents. However, we need to obtain the initial training set with minimum labeling, since the cost to have reviewers to label the document is very expensive.

We applied the *clustering based sample selection* method discussed in section 4.3.1.1 to obtain this initial training set for the active learners. Our target is to have 50 documents for each class in our initial training set. If we pick documents randomly, we might need to label 1000 documents in order to get 50 positive training examples. Instead of doing so, we used the clustering method K-means to categorize the 2500 training documents into two sets. We choose positive examples from the center and boundary from these clusters, since my hypothesis is that the positive examples either appear in the center of the positive cluster and the boundary of the negative cluster. After n positive examples (n equals 10 in our experiment) are obtained, we use these positive examples to create a new cluster. We calculated the center of this positive cluster, and calculated the distance between the rest of the unlabeled training documents and the positive cluster center. After sorting the distance ascending, we picked the positive examples from the head of this queue and pick the negative examples from the tail.

The pseudocode for the *clustering based sample selection* method is as follow:

1. Cluster unlabeled data into two clusters using k-means algorithm.
2. In each cluster, sort every example by their distance to the center of the cluster, and point to the example with the smallest distance
3. While the number of positive examples smaller than $n * 2/3$
   Label the current examples and increment the pointer
4. Point to the example with the largest distance in each cluster
5. While the number of positive examples smaller than $n * 1/3$
   Label the current examples and decrement the pointer
5. Create a new cluster c_{new} with n positive examples, and calculate the cluster centroid $c_{centroid}$
6. Compute and sort the distance $dis_{centroid}$ from all unlabeled examples to $c_{centroid}$
7. Label examples with the smallest $dis_{centroid}$ to obtain the positive training examples
8. Label examples with the largest $dis_{centroid}$ to obtain the negative training examples

In order to testing how much labeled training examples are saved by using this method, we randomly generated 10 different training sets with 2500 documents each. We applied this method to get a training subset with 50 positive examples and 50 negative examples. Table 6.1 reports how many documents are manually labeled to get the training subset in each trial.

	1	2	3	4	5	6	7	8	9	10	Avg.
NO. of Labeled documents	324	321	278	320	335	320	245	260	248	295	294

Table 6.1 The number of documents are manually labeled to get the training subset

The average number of examples that need to be labeled manually to obtain our target training subset is around 295. It's less than 1/3 of the examples that we need to label compared to the random selection. Furthermore, the class distributions for these training subsets are balanced, so we can consider this method as an under-sampling method which does not need all the class labels at the beginning. A small proportion of training examples will be selected by the system to request for the real label, so that less training examples need to be labeled compared to the traditional under-sampling methods.

6.3 Active learning

In the following studies, the active learning algorithm Active-Decorate that has been discussed in section 4.3.2.1 will be applied. Active-Decorate is one of the approaches to implement the Query-by-Committee. Its target is to find the most informative document to be labeled so that when this true label is known, the error rate will be reduced. This document would always be the one that has the largest disagreement between the classifiers in the committee.

The active learning program in this section is developed based on the modified Weka provided by [Melville and Mooney, 2004]. To handle the imbalanced problem, we added the component to generate balanced initial training set for active learners. And we included the document density factor when measuring the utility (see section 4.3.2.1.4).

To find a certain number of extra documents to be labeled, the following procedure is used:

1. Form the committee by creating certain number of classifiers that are diverse.
2. Classify all unlabeled examples with these classifiers.
3. Calculate the utility for all unlabeled examples (disagreement * document density).
4. Choose the N documents that have highest disagreement.

To better understand each parameter we set for this experiment, let us we first look at step 1 closely. Diversity of Classifiers means these classifiers do not tend to agree with each other. At the beginning, the committee only has one classifier, which is the base classifier trained on the initial training set. The next classifier would be trained on the initial training set plus some artificial data generated by the Decorate algorithm according to the Gaussian distribution. The class labels for these artificial examples are set to be the opposite of the current classifier's prediction. For example, if the original classifier predicts an artificial example 30% likely to be positive, Decorate labels this example as 30% likely to be negative. If adding this classifier to the committee decreases the training error, it will be added to the committee permanently, else it will be removed from the committee. The artificial examples will be removed from the training set after a

new classifier is formed in each iteration, no matter this classifier is accepted or not. In this way, we add new classifiers to the committee until the desired number of the classifiers is met or the maximum number of iteration is exceeded. For example, if we want to form a committee with two classifiers, we have to use the artificial data to find another classifier besides the base classifier trained on the initial training set.

When calculating the utility in Step 3, we used Kernel density estimation to calculate the document density. Please refer to details of this algorithm to section 4.3.2.1.4. Document density is the average distance between a given document d and all other documents in the hyper-sphere with radius r, where r is the maximum distance from d to its closest neighbor. Since some of the documents share no attribute with others in our dataset, their r would be very large. We tried to adjust the r by dividing it by a constant α, where $\alpha = 2, 3, 4, 5,$ or 6. According to our experiments, r/4 has the best result.

The parameters setting for this experiment are as follows:

Committee size:	2
Maximum number of iterations for generating committee members:	30
Base learner:	Naïve Bayes
Number of documents chosen to be labeled at each iteration:	2

Table 6.2 The parameter setting for active learning system

In our experiments, we set the committee size to 2. Reported in [MacCallum and Nigam, 1998a], the committee size has little effect to the performance of the active learning system. For the multi-class classification in their studies, they choose a committee size of 3. Since we are working on two-class classification, so we chose a committee size of 2.

We set the number of extra document chosen to label in each iteration to 2. Theoretically, the smaller this number, the better performance of the active learning system. The active learning system will choose the document with the largest

disagreement to be labeled in each iteration. Then this labeled document will be placed into the training set, and this new training set is used to perform the next round of sample selection. Since the training set is changed, the document picked in the next round does not need to be the same as the document with second largest disagreement in the previous round. So if we choose more than one document in each iteration, the performance of active learning cannot be maximized. In our experiment, however, we did not find too much difference between choosing one or two documents in each iteration. But when this number is greater than two, the performance dropped. In order to save computation time while maintain the best result, we set the number of document acquire to label in each iteration to two.

To test the performance improvement by using Active Learning, we had three levels of experiments. The first one used the initial 100 training set we obtained from 6.2 to train the Naïve Bayes classifier. The second one used the initial training set to train the active learners and used these active learners to pick 50 more informative examples from the unlabeled training data to label, and then used these 150 training data to train the Naïve Bayes. The third one is basically the same as the second one, the only difference is using document density factor to weight the document utility when performing active learning. These experimental results are reported in Table 6.3.

	Precision	Recall	F-measure	WSS	Accuracy
Baseline	5%	100%	9.5%	0%	5%
NB	5.9%	100%	11.1%	16.6%	21.65%
NB + AL	9.3%	100%	16.9%	43.4%	48.36%
NB + AL with density	10.5%	100%	19%	49.7%	54.68%

Table 6.3 Results for active learning with BNS and NB

In the first experiment, we used *clustering based sample selection* to obtained a balanced training set and apply it on Naïve Bayes. As most of under-sample methods, we got the very high recall and very low precision in this experiment. Comparing to the under-sampling results in section 5.4, this sample selection method outperforms most of the

under-sampling methods by looking at both precision and recall. Furthermore, this method reduced the labeling effort by 70%.

In the second experiment, the active learning is applied. We used the active learner to iteratively select 50 examples to label and add to the training set. The active learner will choose the example with highest disagreement to request for label. Due to the data imbalance, most of the selected examples are negative examples. So after active learning, the training set is less balanced. However, bringing in these negative examples can help to reduce the ambiguous when classifying the negative class, so that the recall of the negative examples and the precision of the positive examples are increased. After active learning, the precision of the minority class improved to 9.3% and the accuracy improved to 48.36%.

In the third experiment, we involved the document density when calculating the utility of each document in active learning. After adding this knowledge, the precision further improved to 10.5% and the accuracy improved to 54.68%. From the result in table 6.2 you may notice that accuracy improved dramatically when the recall of the minority improved slightly. The precision increased 4.6% from the first experiment to the third experiment; meanwhile the accuracy increased to 33.03%. This large increase in accuracy is due to imbalanced data.

We applied the WSS evaluation that discussed in 4.4.2 to this set of the results because all result with recall higher than 95%. WSS measures the work saved over and above the work saved by simple sampling for a given level of recall. For systematic review, this measurement only makes sense when the high recall is obtained. The WSS value is 49.7% in the third experiment, which means we can save almost half of the human effort in reviewing the documents by using the third method.

We also experimentally investigated how well active learning works with Support Vector Machines or Decision Trees. The results are report in table 6.3 and 6.4.

	Precision	Recall	F-measure	Accuracy
Baseline	5%	100%	9.5%	5%
SVM	5.3%	100%	10%	13.04%
SVM + AL	7.6%	83.9%	13.9%	48.48%
SVM + AL with density	8.4%	79%	15.1%	53.76%

Table 6.4 Results for active learning with BNS and SVM

	Precision	Recall	F-measure	Accuracy
Baseline	5%	100%	9.5%	5%
DT	23.9%	59.7%	34.1%	88.74%
DT + AL	23.5%	51.6%	32.3%	89.28%
DT + AL with density	24.8%	46.8%	32.4%	90.32%

Table 6.5 Results for active learning with BNS and DT

The results of combining Support Vector Machines with active learning (Table 6.3) are similar to the results of combining Naïve Bayes. When using the initial training set to train SVM, we had 100% recall but very low precision and precision increased when adding examples chosen by active learners. The only difference is that SVM cannot maintain 100% recall while improving the precision of the minority class.

By using Decision Trees (Table 6.4), we obtained a very high accuracy with a reasonable recall for the minority class. If both classes weight equally, Decision Trees is a good choice. However, in our study the cost of missing minority examples is very high, so this classifier is not suitable.

In the previous experiments, BNS is used as the feature selection method. According to the experimental result in section 5.3, we understand that the feature ratio affects the performance. In the following experiments, we studied if the feature ratio also influences active learning. We used modified BNS with different feature ratio to obtain different feature sets. Then used the *clustering based sample selection* and active learning to obtain training set and apply it on Naïve Bayes. The results are reported in table 6.6.

#Pos: #Neg	Precision	Recall	F-measure	WSS	Accuracy
Baseline	5%	100%	9.5%	0%	5%
30:330	7.4%	100%	13.8%	27.5%	32.49%
60:300	7.6%	100%	14.1%	30.1%	35.08%
90:270	7.8%	100%	14.50%	35.2%	40.19%
120:240	9.1%	100%	16.7%	41.1%	46.05%
150:210	11.1%	100%	20.1%	53.4%	58.43%
180:180	10.7%	99%	19.3%	49.9%	55.82%
210:150	9.5%	99.8%	17.3%	44.2%	49.41%
240:120	8.2%	99.5%	15.1%	37.7%	43.15%
270:90	8.3%	98.9%	15.3%	37.1%	43.09%
300:60	8.2%	98.2%	16.8%	36.1%	42.73%
330:30	6.4%	98.1%	12%	21.3%	28.13%
360:0	4.2%	98.3%	8.1%	15 %	21.64%

Table 6.6 Results for active learning with Modified BNS and NB

Similarly to the experiments in section 5.3.3, the best result occurs when the feature ratio is 150 to 210. So instead of using the BNS directly to generate a feature select, we can use the *smoothed class distribution function* to calculate the optimal feature ratio and select the minority class features and majority class feature separately by using the modified BNS. This method gives us more flexibility because the class distribution and the cost of the minority class in each case are different.

We also applied the WSS evaluation to this set of the results because all of them are higher than 95%. When we chose the feature ratio as 150 to 210, the WSS value is 53.4%, which means more than half of the workload for labeling documents is saved by applying this method.

In conclusion, we can reduce the labeled training examples and improve the performance by using active learning. In experiments in chapter 5, we randomly select 2500 examples to form the training set. In this chapter, we use the *clustering based sample selection* algorithm to generate the initial training set to train the active learners. By using this algorithm, around 300 examples are required to label in order to get the desired training set. Including the 50 examples selected by active learners to request for label, only 350 examples need to be manually labeled. We save 86% of the labeled examples

by using the active learning compared to 70% saving achieved by using the *clustering based sample selection* alone. Applying our system to the test data, we achieved 100% recall for the minority class and 58.43% overall accuracy. Since we achieved 100% recall, the reviewers only need to go through the documents classified as positive. By achieving work saved over sampling (WSS) as 53.4%, more than half of the human effort in reviewing the documents is saved.

Chapter 7

Conclusions and Future work

7.1 Conclusions

This thesis experimented with different methods to address the bias problems in feature selection, sample selection and classification when the data is highly imbalanced. We also experimented with active learning techniques for imbalanced text classification.

Conclusions on addressing the feature selection bias

We first experimented with the feature selection bias on four popular feature selection methods: IG, Chi^2, Odds and BNS. We found that except BNS which selects features from both classes evenly, IG, Chi^2 and Odds tend to select features from the minority classes. Especially when the number of selected features is small, the proportion of minority class features in the feature sets obtained by IG, Chi^2 and Odds is very large.

We then studied how the bias affects the classification performance. We found that the proportion of minority class features in the feature set influences the recall for the minority class. Since IG selects the largest proportion of minority class features, it outperforms the other feature selection methods in terms of recall. On the other hand, BNS select the features from both classes evenly, so it has highest precision and F-measure, but relative low recall.

To further study how the feature ratio affects the performance of Naïve Bayes, we modified BNS by selecting minority class features and majority class features separately. We experimented with different feature ratios, and found that it is not true that higher the proportion of minority class feature, higher the recall of the minority class. We found the smoothed class distribution function proposed in [Tang and Liu, 2005] works well in estimating the optimal feature ratio by using the class distribution.

Conclusions on addressing sample selection bias

We experimented with five under-sampling methods with Naïve Bayes to address the sample selection bias. Generally, under-sampling the majority class improves the recall of the minority class, but at the same time drops the precision of the minority class. That's because it changed the prior class probability of the training set, which is one of the decision factors in Naïve Bayes. In terms of recall, BNS works well with different under-sampling methods.

Conclusions on addressing classification bias

The skewed data cause Naïve Bayes's decision boundary weights to be biased. We used the Complement Naïve Bayes proposed in [Rennie et al., 2003] to solve this problem. Using Complement Naïve Bayes combined with different feature selection methods improved the recall of the minority class. IG and BNS have higher recall when the number of selected features is small, but when the number of selected features is greater than 500, Chi^2 outperforms the other feature selection methods.

Conclusions on active learning

In this experiment BNS is used as the feature selection method. We used the *clustering based sample selection* algorithm to generate a balanced training set to train the active learner. The result of applying this initial training set directly to Naïve Bayes produced better performance comparing to the result of combining feature selection and under-

sampling. We further expanded the training set by using active learner to select more informative examples to be labeled. By using active learning technique, we saved 86% of the labeled training examples. By achieving work saved over sampling (WSS) as 53.4%, we saved half of the workload for the reviewers. Since we achieved 100% recall for the positive class, the reviewer only need to go through the documents that were classified as positive without worry about missing any relevant documents.

7.2 Future Work

The future work for this thesis is to focus on improvement of the precision of the minority class while maintaining the high recall.

One of the possible future works is to use MeSH to find synonyms for the words in feature set, and expand the feature set by including these synonyms. In English, same meaning can be expressed in different ways by using different words. We may find important words in training set, but miss it in a testing example because different word is used to express the same meaning.

The second possible future works is try to under-sample the training set before performing feature selection. The advantage of this approach is to reduce the feature selection bias. However, the disadvantage is that some important features in the filtered negative examples would be lost. The difficulty of this approach is to find good under-sampling methods so that only the noisy and redundant examples are filtered. This way the loss of the important features can be minimized.

The final possible future works is to study the possibility of using the one-class classification [Zhuang and Dai, 2006]. Actually, our case is very similar to one-class classification problem. We have many negative examples, and these examples are so diverse that they cannot represent the whole class of the negative class. This causes the difficulty to accurately classify the negative class in two-class classification. One- class classification, on the other hand, only uses the positive examples to train the classifier, so we do not need to worry about the diversity and the completeness of the negative class.

Bibliography

[Abe & Mamitsuka, 1998] Abe, N., & Mamitsuka, H., 1998, Query learning strategies using boosting and bagging. Proc. of 15th Intl. Conf. on Machine Learning, *ICML* 1998: 1-10

[Aggarwal and Yu, 2001] Aggarwal C.C., Yu P.S., On Effective Conceptual Indexing and Similarity Search in Text Data. *ICML* 2001: 3-10

[Alexander et al., 2005] Alexander, L.V., Tett S.F.B. and Jonsson T., 2005: Recent observed changes in severe storms over the United Kingdom and Iceland. *Geophys. Res. Lett.,* 32, L13704, doi:10.1029/2005GL022371.

[Alexandersson et al., 2005] Alexandersson J., et al., D5.1 Report on InitialWork in Segmentation, Structuring, Indexing and Summarization, *AMI Annual Report*, 2005

[Bartling et al., 2003] Bartling W.C., Schleyer T.K., Visweswaran S., Retrieval and classification of dental research articles. *Advances in Dental Research* 2003 Dec;17:115-20.

[Blum and Langley, 1997] Blum A. and Langley P., Selection of relevant features and examples in machine learning. *Artificial Intelligence*, 97:245–271, 1997

[Cohn et al., 1996] Cohn, D. A., Ghahramani, Z., & Jordan, M. I. (1996). Active Learning with statistical models. Journal of Articial Intelligence Research, 4, 129-145

[Cohen et al., 2006] Cohen A.M., Hersh W.R., Peterson K., Yen P.Y., Reducing workload in systematic review preparation using automated citation classification. *J Am Med Inform Assoc.* 2006;13:206–219. doi: 10.1197/jamia.M1929

[Craven et al., 1998] Craven M., DiPasquo D., Freitag D., McCallum A., Mitchell T., Nigam K., and Slattery S.. Learning to extract symbolic knowledge from the world wide web. *Proceedings of AAAI,* 1998.

[Domingos, 1999] Domingos, P., MetaCost: A general method for making classifiers cost-sensitive. *Proceedings of the Fifth International Conference on Knowledge Discovery and Data Mining*, pp.155–164, 1999

[Drummond and Holte, 2003] Drummond C. and Holte R.C., C4.5, Class Imbalance, and Cost Sensitivity: Why Under-Sampling beats Over-Sampling. *Proceedings of the Twentieth International Conference on Machine Learning: Workshop on Learning from Imbalanced Data Sets,* 2003

[Frank and Bouckaert, 2006] Frank E., Bouckaert R.R.: Naive Bayes for Text Classification with Unbalanced Classes. *PKDD* 2006: 503-510

[Freund et al., 1997] Freund, Y., Seung, H. S., Shamir, E., & Tishby, N. (1997). Selective sampling using the query by committee algorithm. Machine Learning, 28, 133-168.

[Friedman, 1997] Friedman J. H. 1997. On bias, variance, 0/1 - loss, and the curse-of-dimensionality. *Data Mining and Knowledge Discovery* 1:55–77

[Forman, 2003] Forman G., An Extens ive Empirical Study of Feature Selection Metrics for Text Classification. *Journal of Machine Learning Research 3*: 1289-1305, 2003

[Forman, 2004] Forman G., A pitfall and solution in multi-class feature selection for text classification. *ICML* 2004

[Geman et al., 1992] Geman, S., Bienenstock, E., and Doursat, R., 1992, Neural networks and the bias/variance dilemma. *Neural Computation* 4:1–58.

[Hu et al., 2005] Hu ZZ, Narayanaswamy M, Ravikumar KE, Vijay-Shanker K, Wu CH, Literature mining and database annotation of protein phosphorylation using a rule-based system, *Bioinformatics 21(11): 2759-2765,* 2005

[Japkowicz, 2000] Japkowicz, N. (Ed.), 2000, Proceedings of AAAI'2000 Workshop on Learning from Imbalanced Data Sets. *AAAI Tech Report* WS-00-05.

[Joachims, 1997] Joachims T., A probabilistic analysis of the Rocchio algorithm with TFIDF for text categorization. *ICML*, 1997

[John and Kohavi, 1997] John G., Kohavi R. Wrappers for feature subset selection. AIJ issue on relevance (to appear), 1997

[Kotsiantis and Kanellopoulos, 2006] Kotsiantis, D. Kanellopoulos, P.P., Handling imbalanced datasets: A review, *GESTS International Transactions on Computer Science and Engineering*, Vol.30 (1), 2006, pp. 25-36

[Kubat and Matwin, 1997] Kubat M. and Matwin S., "Addressing the Curse of Imbalanced Training Sets: One Sided Selection," in *Proceedings of the Fourteenth International Conference on Machine Learning*, (Nashville, Tennesse), pp. 179-186, Morgan Kaufmann, 1997

[Lehnert et al., 1995] Lehnert W, et al., Inductive Text Classification for Medical Applications, *Experimental and Theoretical Artificial Intelligence* 7(1), pp. 271-302, 1995

[Lewis and Gale, 1994] Lewis D. and Gale W., A sequential algorithm for training text classifiers. In *Proceedings of ACM SIGIR*, 1994.

[Lewis and Ringuette, 1994] Lewis, D., and Ringuette, M., A comparison of two learning algorithms for text categorization. In *Third Annual Symposium on Document Analysis and IR,* 1994

[Ling and Li, 1998] Ling, C.X., and Li, C., Data mining for direct marketing: Problems and solutions. *Proceedings of The Forth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 73-79. AAAI Press, 1998

[Maloof, 2005] Maloof M., Learning when data sets are imbalanced and when costs are unequal and unknown. In *Workshop on Learning from Imbalanced Data Sets II, ICML*, 2003

[Manevitz and Yousef, 2001] Manevitz L. and Yousef M., One-class svms for document classification. *Journal of Machine Learning Research 2*, pages 139–154, 2001

[McCallum and Nigam, 1998a] McCallum, A., and Nigam, K., Employing EM in pool-based active learning for text classification. *Proc.15th International Conf. on Machine Learning*, pp. 350– 358, 1998. Morgan Kaufmann, CA

[McCallum and Nigam, 1998b] McCallum, A., and Nigam, K., A Comparison of Event Models for Naïve Bayes Bayes Text Classification. *AAAI-98 Workshop on "Learning for Text Categorization".*

[Melville and Mooney, 2004] Prem Melville, Raymond J. Mooney: Diverse ensembles for active learning. *ICML,* 2004.

[Mladenic and Grobelnik, 1999] Mladenic D., Grobelnik M., Feature Selection for Unbalanced Class Distribution and Naive Bayes. *ICML* 1999: 258-267

[Nigam et al., 2000] Nigam, K., McCallum, A., Thrun, S., & Mitchell, T. Text classification from labeled and unlabeled documents using EM. *Machine Learning,* 39, 103–134, 2000

[Nigam, 2001] Nigam K., Using Unlabeled Data to Improve Text Classification. PhD thesis, Carnegie Mellon University, 2001

[Nguyen and Smeulders, 2004] Nguyen H. T., Smeulders A., Active learning using pre-clustering. *ICML* 2004

[Pai et al., 2004] Pai M, McCulloch M, Gorman JD, Pai N, Enanoria W, Kennedy G, Tharyan P, Colford, JM. Systematic reviews and meta-analyses: An illustrated, step-by-step guide. *Natl Med J India* 2004;17(2):86-95

[Rennie et al., 2003] Rennie, J.; Lawrence, S.; Teevan, J.; and Karger, D. Tackling the poor assumptions of Naïve Bayes text classifiers. *Proceedings of ICML,* 2003

[Salton et al., 1989] Salton G., Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer. *Addison-Wesley, Reading, Pennsylvania*, 1989

[Seung et al., 1992]Seung, H. S., Opper, M., & Sompolinsky, H., 1992, Query by committee. Proc. of the ACM Workshop on Computational Learning Theory. Pittsburgh, PA.

[Schohn and Cohn, 2000] Schohn, G., & Cohn, D. Less is more: Active learning with support vector machines. *Proc. 17th International Conf. on Machine Learning*, pp. 839–846, 2000.

[Shultz and Liberman, 1999] Shultz J.M. and Liberman M., Topic Detection and Tracking using idf-weighted Cosine Coefficient, *DARPA Broadcast News Workshop Proceedings*, 1999

[Tang and Liu, 2005] Tang L. and Liu H., Bias Analysis in Text Classification for Highly Skewed Data, ICDM 2005: 781-784

[Tomek, 1976] Tomek, I. Two Modifications of CNN. *IEEE Transactions on Systems, Man and Cybernetics*, SMC-6, 769-772, 1976

[Tong and Koller, 2001] Tong, S., & Koller, D. Support vector machine active learning with applications to text classification. Journal of Machine Learning Research, 2, 45–66, 2001

[Xu et al., 2003] Xu, Z., Yu, K., Tresp, V., Xu, X., & Wang, J. Representative sampling for text classification using support vector machines. *25th European Conf. on Information Retrieval Research, ECIR,* 2003

[Yang and Pedersen, 1997] Yang Y., Pedersen J.O.: A Comparative Study on Feature Selection in Text Categorization. *ICML* 1997, 412-420

[Yu and Liu, 2004] Yu, L. & Liu, H. Redundancy based feature selection for microarray data, in *Proceedings of KDD'04*, ACM Press, New York, NY, USA, pp. 737–742, 2004

[Zenobi & Cunningham, 2001] Gabriele Zenobi, Padraig Cunningham: Using Diversity in Preparing Ensembles of Classifiers Based on Different Feature Subsets to Minimize Generalization Error. ECML 2001: 576-587

[Zhang and Chen, 2002] Zhang, C., & Chen, T. An active learning framework for content-based information retrieval. *IEEE Transaction on multimedia*, Vol. 4, 260–268, 2002

[Zhang and Mani, 2003] Zhang, J. and Mani, I., kNN approach to unbalanced data distributions: A case study involving Information Extraction, *Workshop on learning from imbalanced datasets II, ICML*, 2003

[Zheng et al., 2006] Zheng Z.H., Brady S., Garg A., Shatkay H., Probabilistic Thematic Clustering for Biomedical Text Classification and Feature Selection, *Canadian Student Conference on Biomedical Computing*, 2006

[Zhuang and Dai, 2006] Zhuang L., Dai H.H., Parameter Estimation of One-Class SVM on Imbalance Text Classification. *Canadian Conference on AI* 2006: 538-549