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

Wenhui Li

AUTEUR DE LA THÈSE / AUTHOR OF THESIS

M. (Computer Science)

GRADE / DEGREE

School of Information Technology and Engineering

FACULTÉ, ÉCOLE, DÉPARTEMENT / FACULTY, SCHOOL, DEPARTMENT

Sentiment Analysis: Quantitative Evaluation of Subjective Opinions Using Natural Language Processing

TITRE DE LA THÈSE / TITLE OF THESIS

Prof. Nathalie Japkowicz

DIRECTEUR (DIRECTRICE) DE LA THÈSE / THESIS SUPERVISOR

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

EXAMINATEURS (EXAMINATRICES) DE LA THÈSE / THESIS EXAMINERS

Prof. John Oommen

Prof. Diana Inkpen

Gary W. Slater

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

SENTIMENT ANALYSIS: QUANTITATIVE EVALUATION OF SUBJECTIVE OPINIONS USING NATURAL LANGUAGE PROCESSING

WENHUI LI

THESIS SUBMITTED TO THE
FACULTY OF GRADUATE AND POSTDOCTORAL STUDIES
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE M.Sc. DEGREE IN COMPUTER SCIENCE

SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING
FACULTY OF ENGINEERING
UNIVERSITY OF OTTAWA

Your file Votre référence
ISBN: 978-0-494-50900-5
Our file Notre référence
ISBN: 978-0-494-50900-5

Abstract

Sentiment Analysis consists of recognizing sentiment orientation towards specific subjects within natural language texts. Most research in this area focuses on classifying documents as positive or negative. The purpose of this thesis is to quantitatively evaluate subjective opinions of customer reviews using a five star rating system, which is widely used on on-line review web sites, and to try to make the predicted score as accurate as possible.

Firstly, this thesis presents two methods for rating reviews: classifying reviews by supervised learning methods as multi-class classification does, or rating reviews by using association scores of sentiment terms with a set of seed words extracted from the corpus, i.e. the unsupervised learning method. We extend the feature selection approach used in Turney's PMI-IR estimation by introducing semantic relatedness measures based up on the content of WordNet. This thesis reports on experiments using the two methods mentioned above for rating reviews using the combined feature set enriched with WordNet-selected sentiment terms. The results of these experiments suggest ways in which incorporating WordNet relatedness measures into feature selection may yield improvement over classification and unsupervised learning methods which do not use it.

Furthermore, via ordinal meta-classifiers, we utilize the ordering information contained in the scores of bank reviews to improve the performance, we explore the effectiveness of re-sampling for reducing the problem of skewed data, and we check whether discretization benefits the ordinal meta-learning process.

Finally, we combine the unsupervised and supervised meta-learning methods to optimize performance on our sentiment prediction task.

Acknowledgements

I would like to thank my supervisor Dr. Nathalie Japkowicz for inspiring my interest in Machine Learning and Data Mining, and for guiding me all along through my study and research. Her precious suggestions and criticisms always helped me very much.

Furthermore, I would like to thank Dr. Mohak Shah, who inspired me into the research area of sentiment analysis and gave me many useful references and resources.

I would like to thank Professor Diana Inpken and Professor Stan Szpakowicz, in the whole research, for their bighearted help and patient instructions to this work.

Finally, I must thank all my family members, especially my father and my brother, and thank my best friend Bin Xie, Yaojun Wu, and Guichong Li. They always give generous help to me throughout my Masters studies.

Table of Content

List of Tables

List of Figures

Chapter One

Introduction

If a potential buyer is considering buying a new car, he or she may browse on-line and look for as many as possible related reviews of the brand which interests him or her, for example *Acura*, to get a general idea about whether it is recommended or not by previous customers. Similarly, before watching a movie, many audiences are accustomed to reading reviews to find the opinions of others. This is called *sentiment classification*, in which a document is labeled as a positive ('thumbs up') or negative ('thumbs down') [14] evaluation of a target object such as film, book, manufactured product and so on.

Among the huge amount of favorable and unfavorable opinions toward specific subjects, there has been extensive research on sentiment analysis that classifies texts by 'positive' and 'negative' orientation, such as sentiment classifiers [10,38,39,40,41], affect analysis [42,43], automatic survey analysis [11,14,44,45], opinion extraction [46], or recommender systems [47]. Immediate applications of sentiment analysis include:

- Business intelligence applications, recommender systems and customer relationship management (CRM)
- Filtering "flames" on internet forums and chat rooms
- Labeling hotspot-oriented sentiment to identify and highlight relative semantic polarities of a specific subject, by incorporating sentiment analysis into Information Retrieval and text summarization
- Digitalizing library resources by emotional measurements
- Question answering

At first glance, sentiment analysis is very much like automatic *text categorization*, a kind of topical categorization that attempts to classify documents according to their subject matter. Whereas in traditional text categorization the focus is almost exclusively on the subject matter, such as topic or genre categorization, in sentiment classification the focus is on the assessment of

the writer's sentiment toward the topic. Therefore, sentiment analysis is a kind of non-topical machine learning problem.

In sentiment analysis, because the characterizations are sought of the opinions, feelings, and attitudes expressed in a text, rather than just the facts as for text categorization, the features extracted for the two learning tasks are different.

To date, most work on sentiment analysis has relied on two main approaches. The first ("bag of words") attempts to learn a positive/negative document classifier based on occurrence frequencies of the various terms in the document; within this approach various learning methods can be used to select or weight different parts of a text to be used in classification [10,11,14,16]. The other main approach ("semantic orientation") classifies words (usually automatically) into two classes, "good" and "bad", and then computes an overall good/bad score for the text[12,13,17,20]. This thesis opts for a better solution based on the first approach.

Therefore, the whole sentiment analysis task of this thesis is twofold. First, we investigate how to automatically determine the exact sentiment terms, i.e. features, from the reviews. Second, we explore how to choose and organize machine learning algorithms to classify the reviews according their sentiment. We describe these two stages in detail in section 1.1 and 1.2.

1.1 Stage 1: Feature Selection

We found some foundation of the solution to this problem from earlier work by Hatzivassiloglou and K. R. McKeown, and by Peter D.Turney. Hatzivassiloglou and K. R. McKeown [13] presented the term *semantic orientation* (hereafter SO) which refers to a real number measure of the positive or negative sentiment expressed by a word or phrase. In the following work, the approach taken by Peter D.Turney 2002 [14] is used to derive such values for selected terms in the text. Once the desired terms have been extracted from the text, each one is assigned an SO value. The SO of a term or a phrase is determined based upon the

phrase's pointwise mutual information (PMI) with the words "excellent" and "poor". PMI is defined as follows (1989 Church & Hanks [48]):

$$PMI(word_1, word_2) = \log_2\left[\frac{p(word_1 \,\&\, word_2)}{p(word_1)p(word_2)}\right]$$

While the Semantic Orientation (SO) of a term or a phrase, say *SO(phrase)*, is calculated as follows:

$$SO(phrase) = PMI(phrase, "excellent") - PMI(phrase, "poor")$$

The challenging aspect distinguishing sentiment analysis from traditional topic-based classification is that while topics are often identifiable by keywords frequency alone, sentiment can be expressed in a more subtle manner. In other words, sentiment seems to require more understanding than the usual topic-based classification.

Since Peter D.Turney presented the well known research of "Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews" [14] in 1997, the understanding seemed more plausible and reasonable if the feature selection focuses on adjectives, adverbs, adjectival phrases, and adverbial phrases. Actually, this "*understanding*" of adjective and adverb phrases is frequently used in many previous and after sentiment analysis research, for example, Brill [49], Tony Mullen & Nigel Collier [18], and Vasileios Hatzivassiloglou & Janyce M. Wiebe [16] etc. The basic patterns used to extract phrases are shown in Table 1.1:

	First Word	Second Word	Third Word (Not Extracted)
1	JJ	NN or NNS	anything
2	RB, RBR, or RBS	JJ	not NN nor NNS
3	JJ	JJ	not NN nor NNS
4	NN or NNS	JJ	not NN nor NNS

5	RB, RBR, or RBS	VB, VBD,VBN, or VBG	anything

Table 1.1 Patterns of tags for extracting two-word [14]

In Table 1.1, the JJ tags indicate adjectives, the NN tags denote nouns, the RB tags are adverbs, and the VB tags are verbs. All other tags are derived from the root tags of JJ, NN, RB and VB. For example, RBR tags are comparative adverb, RBS tags are superlative adverb, NNS tags are plural noun, VBD tags are the past tense of auxiliary verb *be*, VBN tags are past participle of auxiliary verb *be*, and VBG tags are present participle of auxiliary verb *be*.

Due to the good performance of Turney's method [14], his PMI-IR measure and SO estimations have been accepted and widely adopted as an effective measure for feature selection.

The questions we are asking here are: is it possible that some other potentially beneficial terms, in topic categorization research, are out of our sight when excessive emphasis is made on the difference between sentiment analysis and traditional topical categorization? Is it kind of hypercorrection using only adjectives and adverbs as sentiment terms (features) for sentiment analysis too extreme? What should be thoughtfully noted is that Peter D. Turney also mentioned: "The latter difficulty (the limitation of PMI-IR method) might be addressed by using semantic orientation combined with other features in a supervised classification algorithm" [14] (page 8 of 8).

What kind of other features may contribute to sentiment analysis? Actually, Most of the words used in customer reviews are nouns, verbs, adjectives, and adverbs, which are named substantive. However, in most of previous research, they focus the feature selection on adjectives and adverbs only. How about nouns and verbs? Although nouns and verbs do not directly indicate customers' mood in the reviews, they semantically correlate with the objects or actions that can express their happiness or sadness or indicate the attitude of people who serve them. Take the top 10 most frequent words from the corpora of bank reviews extracted from Epinions web site (http://www.epinions.com[27]) as an example, as shown in Table 1.2:

	Positive		Negative	
	Words	Frequency	Word	Frequency
Nouns	protection	216	complaint	236
	friend	262	nightmare	116
	advantage	134	need	329
	offer	286	chase	184
	trust	123	bad	131
	experience	722	problem	1490
	care	209	error	354
	deal	256	terribleness	80
	good	128	fraud	96
	free	375	cost	316
Verbs	offer	1013	hate	121
	clear	411	owe	122
	experience	156	drop	126
	hope	174	hit	170
	resolve	174	need	1418
	contact	208	steal	170
	understand	298	cut	93
	care	310	bother	132
	consider	310	refuse	233
	save	409	cost	233

Table 1.2 Top 10 positive and negative nouns and verbs

(Excerpt from 20 top-ranked sentiment words set)

In Table 1.2, the noun words such as 'advantage', 'trust', 'care', and 'free' possibly are positive factors for customers' good mood; on the other hand, 'complaint', 'nightmare', 'problem', and 'fraud' are very likely to contribute to negative opinions; Verbs, though, express the unsatisfactory opinion in a implicit way, for example, 'offer' usually is used to construct phrases such as 'offer free cheque book', 'offer 7*24 hotline service', or 'offer reward points', while 'refuse' is often related to customers' complaint about the bad attitude of bank employees.

Relying on the PMI-IR method or extracting sentiment words from a prepared fixed dictionary

is not only insufficient but also unreliable, especially for capturing sentiment nouns and sentiment verbs. Fortunately, WordNet provides semantic relatedness information between concepts including nouns and verbs, so it is a potentially useful resource for indicating the mood level of nouns and verbs by checking the semantical and syntactical relations between a noun or a verb and any other part of speech. Using WordNet not only helps the PMI-IR method improve the selection of sentiment terms, but also the absence of some important sentiment terms from the General Inquirer dictionary. Furthermore, because WordNet provides relatedness measures between adjectives and between adverbs, it makes the selection of adjectives and adverbs more reliable than using PMI-IR approach only.

1.2 Stage 2: Unsupervised learning and supervised learning

In this stage, the bank reviews will be classified into different classes by supervised learning; while by unsupervised learning, we are trying to scoring reviews using scores as close as that given by the customers.

As described in Stage 1, previous works miss important aspects of the quantitative evaluation such as the commonly used five star review rating system. On many customer review websites, it is usual to define one star as "poor" and five stars as "excellent", which is the reason why Turney's research used "poor" and "excellent" as the reference words for PMI-IR. This five star review rating system is as follows:

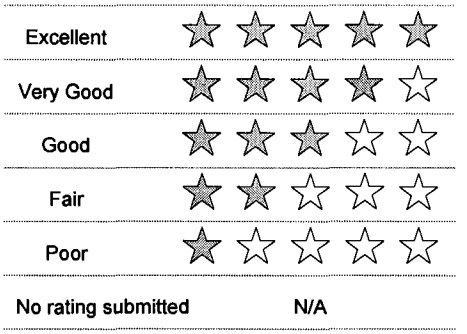

Fig 1.1 Five Star Rating Scheme

Labeling these articles with their sentiment would provide succinct evaluation to readers; indeed, these labels are part of the appeal and value-added of such sites as www.consumerreports.org, which both labels customer reviews that do not contain explicit rating indicators and normalizes the different rating schemes, which individual reviewers use, into the five star rating schemes.

The personal, evaluation-like nature of reviews indicates not only reviewers' mood polarity at the time of posting, but also the degree to which they like or dislike a product or service. For example, for bank reviews, both "Washington Mutual" and "Bank of America" are classified as "good" by a binary classifier. Obviously, this kind of labeling does not give a clear recommendation regarding which one is a better candidate; whereas customers expect an intensity indicator of different sentiments among reviewers. In other words, binary classification is too limited to suggest the level of satisfaction.

In this thesis, we address the task of estimating this state-of-favorability from the text written by reviewers. To this end, we build models that try to predict the levels of various moods according to the language used by reviewers; our models show high correlation with the moods actually measured, and substantially outperform a baseline, which uses only PMI-IR for feature selection.

The task of classifying each individual review with respect to its favorability has been shown

to be a very difficult task by Mishne 2005 [50]. Our goal is to look for the possible approaches to improve the accuracy of the classification. Standard classification algorithms cannot use the ordering information because they treat the class attribute as a nominal quantity, in other words, a set of unordered values. However, in the scenario of a five star rating system, the class values exhibit a natural order which may greatly benefit predicting the degree of sentiment orientation.

Our hypothesis works on a simple approach to ordinal classification (Eibe Frank and Mark Hall 2001) which attempts to enable classification algorithms to make use of the ordering nature of five star grading schemes. Because this method benefits tree learners, an essential learner in our experiments, we will discuss whether it outperforms the Bayesian classifiers such as Naïve Bayes and BayesNet, and explore the performance of Support Vector Machines (SVM hereafter).

Most previous works, including Michael Gamon and Anthony Aue [17] and Peter D.Turney [14], point out that sentiment analysis is closely related to the domain of topic categorization. Due to the highly domain-specific nature of the sentiment classification task, moving from one domain to another typically requires the acquisition of a new set of training data, and the feature set must be manually added to a special lexicon and manually tagged to indicate positive or negative sentiment for binary classification.

Many previous works adopt movie reviews as their testbed, but this thesis is based on customer review data of forty six banks from www.epinions.com; because, according to Peter D.Turney [14], the review of services, such as the review of banks, presents a higher correlation between the average semantic orientation and the number of stars assigned by the author. Of course, the feature set is specific to the domain of bank reviews, and must be re-built anew for each new domain if the domain is changed. However, this thesis is still instructive for open-domain sentiment analysis, even if it uses a domain-specific data source.

During the study, we encountered the class imbalance problem in our experiments. We investigated and explored re-sampling and discretization to minimize the effect of the bias

towards the majority class. Furthermore, we found that the unsupervised learning and supervised learning can complement each other, so we combined them and obtained better learning performance.

To conclude, our work directly addresses both issues of feature selection and multiclass classification for quantitative sentiment scoring. From this discussion and in the following chapters, we can see that both modifications, i.e. incorporating WordNet into sentiment feature selection and enabling the use of ordinal information, effectively enrich the feature sets and significantly improve the learning results. In addition, when delt with the problem of imbalanced data distribution by combining supervised learning and unsupervised learning, we found that this approach compensate for the weakness of the both kind of learners.

1.3 Main Contribution

The first contribution of this thesis is that it combines unsupervised learning and typical supervised learning algorithms to implement quantitative sentiment analysis via a multi-class classification approach.

As far as we know, most of the previous sentiment analysis research has focused on binary sentiment classification problems, but almost none of it attempts to implement quantitative sentiment analysis.

This thesis focuses on the quantitative evaluation of subjective opinions, and achives good performance for the five star scoring task. In this thesis, we widely explore diverse unsupervised learning and supervised learning algorithms, adopt various input and output engineering methods to improve the learning performance, and finally contribute an effective approach that shares the advantages of different models.

The second contribution of this thesis is that we explore the effect of various types of features,

and via meticulous experiments and comparisons, we propose our own SO_WN measure that effectively enhances the quality of feature sets.

We exhaustively experiment with unigrams, bigrams including valence shifters, SO-PMI-IR selected features, WordNet selected features, and WordNet derived synthetic semantic features, compare their performance using four different learning algorithms, and finally discover that WordNet selected features and WordNet derived synthetic semantic features both significantly improve the learning result. In other words, the semantic and syntactic information provided by WordNet is a promising resource for feature selection.

Thirdly, this thesis adopts a simple meta-learning solution to make use of ordering information of five star score labels, and uses the ordinal meta-classifier to produce relatively better experimental results than the traditional multiclass classification algorithms which treat the score of reviews as a nominal quantity.

The fourth main contribution of this thesis is that we propose a creative approach to share the advantages of unsupervised and supervised learning by combining their output according to the class distribution of the imbalanced dataset.

When dealing with the problem of imbalanced data distributions, we find that supervised learning is apt to excessively bias toward majority classes; conversely, unsupervised learning is not as affected by the skewed datasets because it is more natural for capturing the sentiment contained in natural language. Therefore, we combine unsupervised learning and supervised learning to learn from strong points of both methods and offset their weaknesses. This method obtains significant improvement on the skewed data of all the 3164 bank reviews. This promising approach of output engineering can greatly benefit the problems of imbalanced data distribution during multiclass classification.

1.4 Thesis Layout

The second chapter attempts to establish an exhaustive literature review of previous works in the sentiment analysis area and also discusses in detail their important conclusions and ideas. In addition, Chapter 2 describes related resources, for example WordNet, and presents their merits and contributions.

Chapter 3 is the preliminary study in which we implement some prerequisite trial experiments and present the heuristics of our methodology.

Chapter 4 explains our methodology for both supervised and unsupervised learning methods, the mechanism of combining them, and discusses the details of the experimental process.

Chapter 5 demonstrates our ordinal meta-learning approach for the multi-class classification task and unsupervised learning with WordNet measures, analyzes their experimental results, and attempts to improve the performance by combining standard classifiers with our unsupervised learner.

Chapter 6 provides a concludion of this thesis and proposes several promising directions for future work.

Chapter Two

Background

Chapter 1 provided an overview of sentiment analysis and introduced the motivation, purpose and machine learning approach of this thesis. This chapter discusses related background knowledge required for this thesis.

In section 2.1 and section 2.2, we discuss text categorization and sentiment analysis respectively, and present the relationship and difference between them.

In section 2.3, we will look at the lexical resource, WordNet, which plays an important role in our feature selection process. WordNet is chosen because of its abundant approaches for measuring the relationship between words. It is a commonly used resource in Natural Language Processing (hereafter NLP). The words in WordNet are organized into the hierarchy according to the relation between each word. By understanding the internal structure of WordNet, it can be better used in the developing of review rating methods.

In section 2.4, we review the machine learning algorithms, including Naïve Bayes, BayesNet, C4.5 decision tree and SVM, which are used in this thesis.

Lastly, from section 2.5 to section 2.9, we will introduce the natural language processing resources and tools used in our experiments. These resources include the GPoSTTL Tagger/Lemmatizer, the syntactical parser Linksys, the WordNet::Similarity package for estimating the semantic relatedness of sentiment concepts, the dictionary General Inquirer, and the stop words list.

2.1 Text categorization

Due to the increased availability of digital documents and the ensuing requirement to access them in flexible ways, content-based document management tasks, known as Information Retrieval (hereafter IR), have been dramatically increasing in importance. Text categorization, the activity of labeling natural language texts with thematic categories from a predefined set, is one such task.

Text categorization is now being applied in many contexts, ranging from document indexing based on a controlled vocabulary, to document filtering, automated metadata generation, word sense disambiguation, population of hierarchical catalogues of Web resources, and in general any application requiring document organization or selective and adaptive document dispatching.

In general, text categorization is a supervised learning process. Using a labeled training set of documents, a data representation model could be created between the features and classes of the documents. Assuming that C is the set of class labels and D is the training set of documents, then there exists an objective concept T:

$$T : D \rightarrow C. \qquad \textbf{(2.1)}$$

Here, T maps a sample document to a certain class. For every document d in set D, $T(d)$ is determined. Via a supervised learning process on the *training set*, a data representation model H approximate to T is found:

$$H : D \rightarrow C. \qquad \textbf{(2.2)}$$

That is, for each new document d_n, $H(d_n)$ represents the classified result of d_n. The goal of supervised learning is to find a maximum approximate H of T. In other words, given an evaluation function f, the aim of learning is to minimize the difference between T and H as described in formula 2.3:

$$Min(\sum_{i=1}^{|D|} f(T(d_i) - H(d_i)). \qquad \textbf{(2.3)}$$

Typically, text categorization includes five steps, as shown in Figure 2.1:

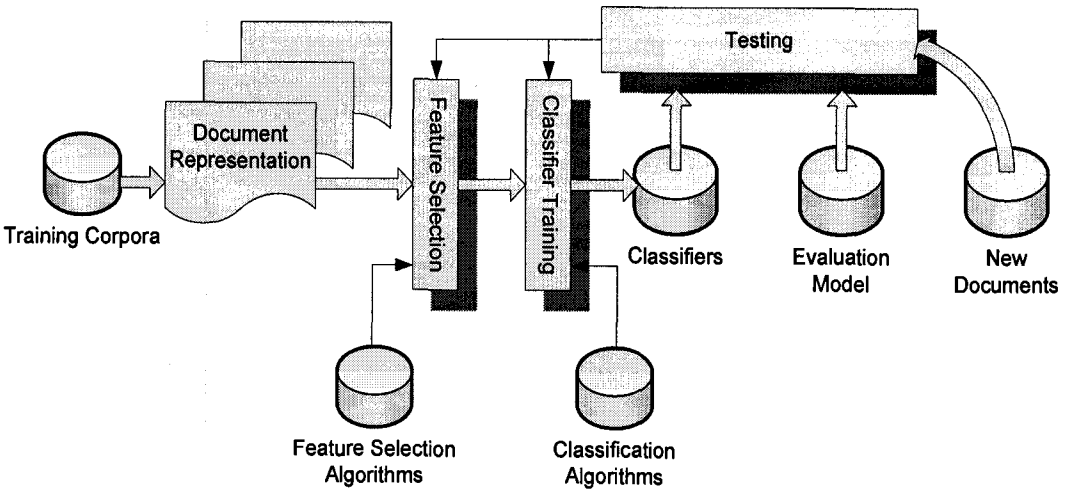

Fig. 2.1 The process of text categorization

1. Collecting the *training set* of documents

   The first step in text categorization is to transform documents into a training set suitable for the learning algorithm and the classification task. This training set should represent and cover documents in every class. In general, the training set is a manually generated corpus.

2. Creating the representation model

   This step involves selection of linguistic elements from the documents and mathematical representation of these elements. For text categorization, the information retrieval vector space model is frequently used as the data representation. Each object in the training set is represented in the form $(\vec{x}, c)$, where $\vec{x} \in \Re^n$, is a vector of measurements and c is the class label. Consequently, the training set of corpus can be represented as shown in Figure 2.2 and 2.3.

$$\vec{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{pmatrix}$$

Fig. 2.2 A vector $\vec{x}$ of a document

n is the number of features, x_i donates i^{th} component of $\vec{x}$ (its value on dimension i)

$$A = \left[\begin{array}{c|cccccccc} & W_1 & W_2 & W_3 & \cdots & W_{N-2} & W_{N-1} & W_N & \\ \hline D_1 & 0 & 0 & 1 & \cdots & 0 & 1 & 0 & L_1 \\ D_2 & 1 & 0 & 0 & \cdots & 0 & 0 & 1 & L_2 \\ D_3 & 0 & 0 & 1 & \cdots & 0 & 0 & 0 & L_3 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \\ D_{M-2} & 1 & 0 & 1 & \cdots & 1 & 0 & 0 & L_{M-2} \\ D_{M-1} & 0 & 1 & 0 & \cdots & 0 & 0 & 1 & L_{M-1} \\ D_M & 0 & 1 & 1 & \cdots & 0 & 0 & 0 & L_M \end{array}\right]$$

Fig. 2.3 An example of a term presence matrix A

(D: Document, W: Word, F: Feature, L: Label)

3. Feature selection

Language is an open system; thus, either the printed or electronic document, as a representation of language, is open too. This openness results in an infinite number of of candidate features. Therefore, the classification system should choose as few as possible features which are closely related to the classification attribute.

4. Choosing a classifier

In text categorization, classifiers take the responsibility of mapping relationship from feature vectors to classes of topics. so choosing the proper classifier is a kernel problem. The commonly used methods include Naive Bayes [3, 4, 5], KNN [6], Regression Model [7], Support Vector Machine [8, 9] etc. Among those approaches,

Naive Bayes and SVM perform better than other algorithms, and hold considerable stability.

5. Performance evaluation model

A good evaluation model which can measure classifiers' intrinsic performance could be used as the target function to improve the classifiers. In text categorization, which evaluation parameters are used depends on the type of classification. Binary classification and multi-class classification adopt different evaluation parameters. Currently, precision, recall, and accuracy are most commonly used in IR (Information Retrieval).

2.2 Sentiment Analysis

The technique to determine favorable and unfavorable opinions toward specific subjects within large volumes of documents is called sentiments analysis. The problem of detecting the sentiment of documents has often been approached via highly human-structured or very complex methods. These approaches have generally adhered to splitting an input corpus into two categories, positive and negative. In other words, so far, most sentiment recognition tasks concentrate on binary classification. Sentiment classification research is similar to but quite different from general tasks of text categorization.

There are a number of challenging aspects of sentiment analysis. Opinions in natural language are very often expressed in subtle and complex ways, presenting challenges which may not be easily addressed only by simple text categorization approaches such as *n-gram* or keyword identification approaches. Although such approaches have been employed effectively (Pang et al., 2002), there appears to remain considerable room for improvement.

As a relatively new area, sentiment classification is related but different with topic categorization. Many researchers have been utilizing the methods used by topic-based categorization on sentiment classification, because sentiment, like genre, tends to be treated as an attribute of documents. Consequently, they experiment with traditional methods inherited from text categorization. Intuitively, the representation model of sentiment is learned from essential units: words, phrases and n-grams, but the learning process has its own difficulties. These difficulties include:

1. Valence shifters change the original meaning of terms

   The issue of sentiment classification is challenging when compared to that of topic categorization. In topic categorization, the labels rely heavily on observing so-called key words, e.g., 'mp3 player', 'transportation', most of which are nouns implying the meaning of a topic. (Interestingly here, unigrams perform well in topic categorization, and likewise not bad in sentiment classification [10]). Some previous works thought that a similar strategy might work for sentiment categorization—simply look for words like 'horrible', 'great' (usually be adjectives), and classify accordingly. However, many valence shifters and negations change the degree of those adjective terms or even reverse their meanings. [11]

2. Implicit semantic orientation

   Counter-intuitively, semantic orientation does not necessarily associate with terms of polarity; the favorability could be expressed via implicit ways. For example, there are no positive words in "I admit it's a really awful loan option ... such a low interest..." which is from a bank review, but implies strong positive evaluation, even though 'awful' generally holds negative polarity. Likewise, "How could customers bear such kind of attitude?" has no negative words, but presents extreme dissatisfaction.

3. The calculation cost of large sets of features

There is no clear ultimate conclusion as to what features are most efficient for sentiment classification. Moreover, sometimes unexpected words, such as "still", or even punctuation contribute to semantic orientation. Since any feature cannot be simply ignored, the cost of running the machine learning algorithm rises.

4. Multi-subjects/topics

For topic classification, the whole document is classified according to whether it belongs to one topic. Unfortunately, it does not hold for sentiment analysis. In one subjective review collected from the Internet, many aspects or objects might be discussed about products or services. For instance, a review of a movie often involves the actress/actor, directors, or cameraman etc., and the opinions are about every part of a movie but not the whole. Thus, how to extract the opinions for each unit in a document and accurately associate sentiment to a specific object is definitely a challenging job.

5. Challenges of Unsupervised learning.

For unsupervised learning, the granularity of a sentiment unit is a key factor of sentiment analysis. Looking at the relationship between words/phrases, sentences and articles is an attractive research direction. Obviously, in one document, the sentiment orientation of phrases, sentences and the whole article are rarely always consistent, so they have to be thought about separately. Therefore, how to build a hierarchical system to effectively organize the sentiment carried by different layers of documents determines the final accuracy of quantitative scoring of the corpus.

Sentiment analysis is a domain-specific problem instead of open-domain research; In other words, sentiment analysis is sensitive to content, genre, and style. Because different styles, genres, and contents present diversified linguistic preference, the feature selection should

adopt different strategies. For example, in news reports, verbs and their inflected forms are more important than other parts of speech, while in customer reviews using all substantives (adjectives, adverbs, nouns and verbs) and bigrams usually yields better classification result.

Although sentiment analysis differs from text categorization, they are closely related to each other. Previous research by Tony Mullen and Nigel Collier [18] shows that topic-annotated dataset can produce better performance than non-topic-annotated dataset; on the other hand, the sentiment classification result benefits topic classification as well. That is, sentiment and topic are correlated to each other.

2.3 The Lexical Database - WordNet

WordNet is an electronic lexical database. It is considered, by researchers in computational linguistics, text analysis, and many related areas, to be the most important available resource. Due to its rich semantic and syntactical information about words, we adopt it as an important measure and source in our feature selection process. The design of WordNet is inspired by current psycholinguistic and computational theories of human lexical memory. English nouns, verbs, adjectives, and adverbs are organized into synonym sets (hereafter synset), each representing one underlying lexicalized concept. Different relations link the synonym sets.The WordNet system includes four parts shown as follows:

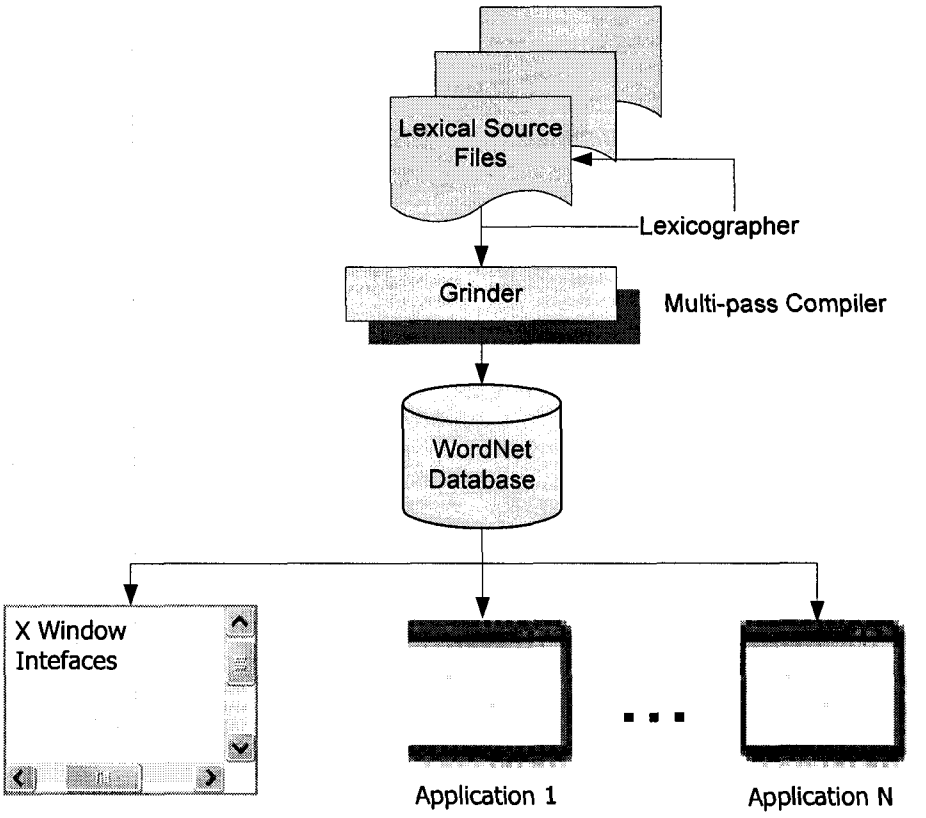

Fig 2.4 The Structure of WordNet

1. The WordNet lexicographers' source files;
2. The software, Grinder, to convert lexical source files into the WordNet lexical database;
3. The WordNet lexical database;
4. The suite of software tools used to access the database.

WordNet organizes nouns, verbs, adjectives and adverbs into synsets, which are further arranged into a set of lexicographers' source files by their syntactic category and other organizational criteria. Adverbs are stored in one file, while nouns and verbs are grouped according to semantic fields. Adjectives are divided between two files according to their types: one for descriptive adjectives and one for relational adjectives.

Each source file contains a list of synsets for one part of speech. Each synset comprises synonymous word forms, relational pointers, and other information. The relations denoted by

these pointers include (but are not limited to): hypernymy/hyponymy, antonymy, entailment, and meronymy/holonymy. Figure 2.5 describes the hypernymy/hyponymy structure of word 'chair', in which word 'chair' has four different word senses:

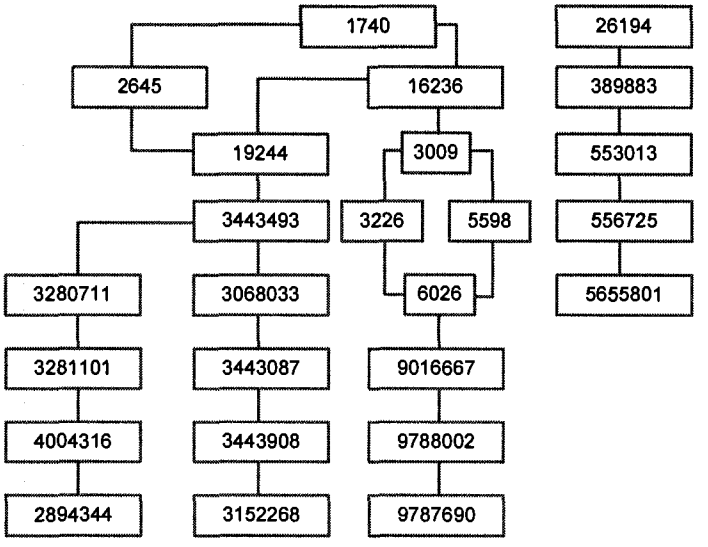

Fig 2.5 The concept chain of word 'chair'

In Figure 2.5, each rectangle is a synset, while the number is the key index of the synset in WordNet. For example, the chain

2894344<4004316<3281101<3280711<3443493<19244<2645^16236<1740

represents:

{chair}<{seat}<{furniture piece_of_furniturearticle_of_furnitrure} <{furnishings} <{instrumentality instrumentation}<{artifact artefact}<{object physical_object}^{whole whole_thing unit}<{entity}.

Polysemous word forms are those that appear in more than one synset, therefore representing more than one concept. A lexicographer often enters a textual gloss in a synset, usually to provide some insight into the semantics intended by the synonymous word forms and their usage. At present, the textual gloss is included in the database and can be displayed by retrieval

software. Comments can be entered, outside of a synset, by enclosing the text of the comment in parentheses, and are not included in the database.

As shown in Figure 2.4, the Grinder tool compiles the lexicographers' files, and organizes them via the relational pointers into the WordNet database. Relational pointers represent the relations between different word forms in a synset and other synsets, and are either lexical or semantic. Table 2.1 summarizes the relational pointers by syntactic category.

Noun		Verb		Adjective		Adverb	
Antonym	!	Antonym	!	Antonym	!	Antonym	!
Hyponym	~	Troponym	~	Similar	&	Derived from	\
Hypernym	@	Hypernym	@	Relational Adj.	\		
eronym	#	Entailment	*	Also See	^		
Holonym	%	Entailment	>	Attribute	=		
Attribute	=	Also See	^				

Table 2.1 WordNet Relational Pointers

In WordNet, a word form is represented as the orthographic representation of an individual word or a string of individual words joined with underscore characters. A string of words so joined is referred to as a collocation and represents a single concept, such as the noun collocation presidential_election. Each word form in WordNet is known by its orthographic representation, syntactic category, semantic field, and sense number. Together, these data make a primary key uniquely identifying each word form in the WordNet relational database.

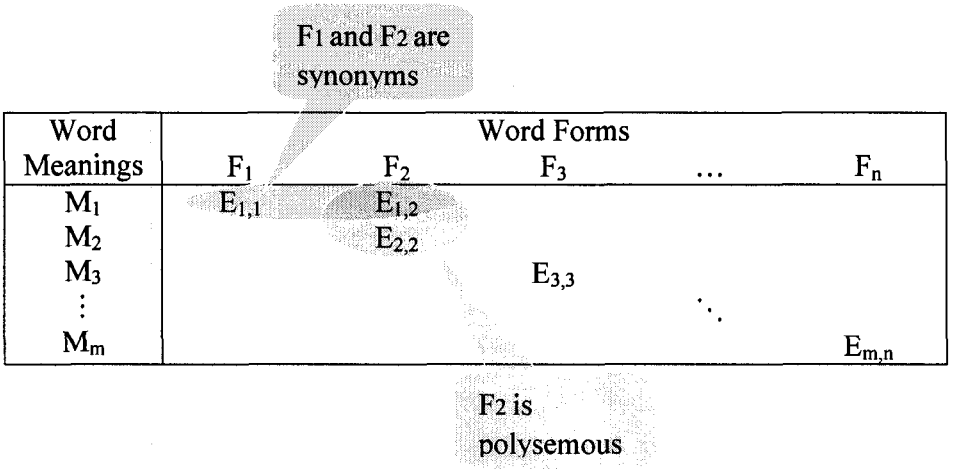

Table 2.2 Illustrating the Concept of a Lexical Matrix

Table 2.2 illuminates the notion of a lexical matrix in detail. Word forms are viewed as headings for the columns (denoted by the uppercase letter F); word meanings as headings for the rows (denoted by the uppercase letter M). An entry in a cell of the matrix implies that the form in that column can be used (in an appropriate context) to express the meaning in that row (denoted by the uppercase letter E). Thus, entry $E_{1,1}$ implies that word form F_1 can be used to express word meaning M_1. If there are two entries in the same column, the word form is polysemous; if there are two entries in the same row, the two word forms are synonyms (relative to a certain context).

As this thesis is written, the current version of WordNet is version 2.1. It includes about 110,000 noun word forms organized into approximately 75,800 word meanings (synsets). The numbers are approximate because WordNet continues to grow. Many of these nouns are compounds, in which a few are artificial collocations, phrasal verbs, and idiomatic phrases invented for the convenience of categorization. At the same time, the relationships including synonymy/antonymy, hypernymy/hyponymy/trononymy, meronymy/holonymy and entailment are maintained together with those word forms together. In addition, some simple syntactical information, such as verb tense frames are also included.

From 1978 to present, WordNet has been developing rapidly. Interestingly, WordNet has been already gradually ignored by phycholinguists, while far more interesting ways shown by computational linguists. The reason behind this phenomenon is that WordNet is organized hierarchically and conceptually, and that hierarchical families of lexicalized synsets actively accelerated automatic recognition of natural language processing. For example, the ontological information hidden in noun-, adjective-, and adverb-hierarchies explains much more semantic facts than researchers' imagination, and thereby benefits our study in this thesis, for determining the distance between word senses.

2.4 Classifiers

2.4.1 Support Vector Machine

Support Vector Machines (SVM) are learning systems that use a hypothesis space of linear functions in a high dimensional feature space, trained with a learning algorithm from optimization theory that implements a learning bias derived from statistical learning theory. This learning strategy introduced by Vapnik et al. is a principled and very powerful approach that in recent years has outperformed most other systems in a wide variety of applications.

The goal of Support Vector Machines is to design a computationally efficient way of learning 'good' separating hyperplanes in a high dimensional feature space. In this definition, the 'good' hyperplanes mean SVMs that can optimize the generalization bounds, and 'computationally efficient' denotes that these SVM algorithms should be able to deal with sample sizes of the order of 100 000 instances in a reasonable amount of time. Generalization theory gives guidance about how to control capacity and hence prevent overfitting by controlling the hyperplane margin measures, while optimization theory provides the mathematical techniques necessary to find hyperplanes optimizing these measures. Different generalization bounds exist, and inspire different algorithms separately, e.g. optimizing the maximal margin, the margin distribution, the number of support vectors, etc.

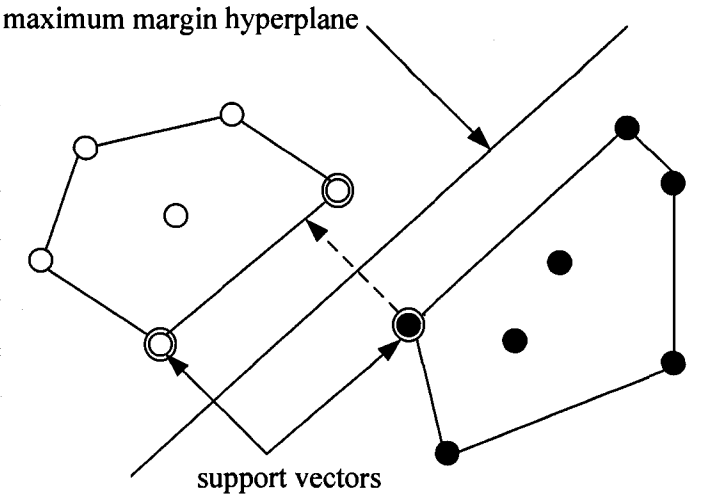

Fig 2.6 A maximum margin hyperplane

The simplest model of Support Vector Machine is the so-called maximal margin classifier, as shown in Figure 2.6. It works only for data which are linearly separable in the feature space, and hence cannot be used in many real-world situations. Nonetheless it is the easiest algorithm to understand, and it forms the main building block for the more complex Support Vector Machines. This maximal margin strategy exhibits the key features that characterize this kind of learning machine, and its description is therefore critical for understanding the more advanced systems.

The maximal margin classifier optimizes these generalization bounds by separating the data with the maximal margin hyperplane, and given that the bound does not rely on the dimensionality of the space, this separation can be discovered in any kernel-induced feature space. The maximal margin classifier forms the strategy of the first Support Vector Machine, namely to find the maximal margin hyperplane in an appropriately chosen kernel-induced feature space.

As shown in Figure 2.6, the instances which are closest to the maximum margin hyperplane, holding a minimum distance to it, are called *support vectors.* There is always at least one

support vector for each class, and there are often more. The important thing is that the set of support vectors uniquely defines the maximum margin hyperplane for the learning problem. Given the support vectors for the two classes, we can easily construct the maximum margin hyperplane. All other training samples are irrelevant, because they can be deleted without changing the position and direction of the hyperplane. A hyperplane separating the two classes could be written as:

$$v = b + w_0 + w_1 a_1 + w_2 a_2 + \ldots + w_n a_n$$

In the above formula, the a_i is the numeric attributes for each example, while w_i is the weights learned by the algorithm to allow for the greatest separation of the two classes. The best hyperplane is therefore the one that minimizes the average error between the points and itself. Here, for SVMs, the function used to classify the instances is totally different from linear regression.

The above equation could be written in another form. When using linear regression as a binary classifier, we can force the function output to values of ± 1. Then the maximum margin hyperplane can be denoted as:

$$v = b + \sum \alpha_i y_i a(i) \cdot a$$

Here, i is a support vector. The term $a(i) \cdot a$ represents the dot product of the test instance with one of the support vectors. From another angle, the *a(i)* may be thought of as the whole set of attribute values for the i th support vector. y_i is a hyper parameter used for optimization when we generating the linear model and doing pattern search. Finally, b and a_i are parameters that determine the hyperplane, just as the weights w_0, w_1, and w_2 are parameters that determine the hyperplane in the earlier formulation.

2.4.2 Naïve Bayes

Naïve Bayes is a simple and intuitive method based on Bayes's rule of conditional probability.

This method goes by the name of Naïve Bayes, because it's based on Bayes's rule and "naïvely" assumes independence-it is only valid to multiply probabilities when the events are independent. The assumption that attributes are independent (given the class) in real life certainly is a simplistic one. But despite the disparaging name, Naïve Bayes works very well on actual datasets, particularly when combined with some of the attribute selection procedures that eliminate redundant, and hence nonindependent, attributes.

Naïve Bayes gives a simple approach with clear semantics, for representing, using, and learning probabilistic knowledge. Impressive results can be achieved using it. It has often been shown that Naïve Bayes rivals, and indeed outperforms, more sophisticated classifiers on many datasets.

Repeatedly in machine learning people have eventually, after an extended struggle, obtained good results using sophisticated learning methods only to discover later that simple methods such as 1R, the simple method using a single rule, and Naïve Bayes do just as well—or even better.

2.4.3 Bayesian Network

A Bayesian network is a probabilistic graphical model that represents a set of variables and their probabilistic independencies. The term "Bayesian networks" was coined by Pearl (1985) to emphasize three aspects:

1. The often subjective nature of the input information.
2. The reliance on Bayes's conditioning as the basis for updating information.
3. The distinction between causal and evidential modes of reasoning, which underscores Thomas Bayes's paper of 1763.[74]

In this thesis, the Bayesian networks are used to represent the probabilistic relationships between sentiment score and features (such as presence or frequency of sentiment phrases). Given features, the network can be used to compute the probabilities of the result of five scores

from1 to 5.

Suppose $n_1, n_2, \ldots, n_k$ is the number of times word i occurs in the reviews we collect, and P_1, $P_2, \ldots, P_k$ is the probability of obtaining word i when sampling from all the reviews in category H. Assume that the probability is independent of the word's context and position in the review. These assumptions lead to a *multinomial distribution* for review probabilities. For this distribution, the probability of a review E given its class H—in other words, the formula for computing the probability Pr[E|H] in Bayes's rule—is

$$\Pr[E \mid H] \approx N! \times \prod_{i=1}^{k} \frac{P_i^{n_i}}{n_i!}$$

where $N = n_1 + n_2 + \ldots + n_k$ is the number of words in the review. The reason for the factorials is to account for the fact that the ordering of the occurrences of each word is irrelevant according to the bag-of-words model. P_i is estimated by computing the relative frequency of word i in the text of all training reviews pertaining to category H. In reality there should be a further term that gives the probability that the model for category H generates a review whose length is the same as the length of E (that is why we use the symbol a instead of =), but it is common to assume that this is the same for all classes and hence can be dropped [59].

2.4.4 C4.5 Decision Tree

C4.5 decision tree is developed from the ID3 decision tree. A series of improvements to ID3 culminated in a practical and influential system for decision tree induction called C4.5. These improvements include methods for dealing with numeric attributes, missing values, noisy data, and generating rules from trees.

C4.5 builds decision trees from a set of training data in the same way as ID3, using the concept of Information Gain. The training data is a set $S = s_1, s_2, \ldots$ of already classified samples. Each sample $s_i = x_1, x_2, \ldots$ is a vector where $x_1, x_2, \ldots$ represent attributes or features of the sample. The

training data is augmented with a vector $C = c_1, c_2, ...$ where $c_1, c_2, ...$ represent the class that each sample belongs to.

C4.5 uses the fact that each attribute of the data can be used to make a decision that splits the data into smaller subsets. C4.5 examines the normalized Information Gain (difference in entropy) that results from choosing an attribute for splitting the data. The attribute with the highest normalized information gain is the one used to make the decision. The algorithm then repeats on the smaller sublists.

This algorithm has a few base cases, and the most common base case is when all the samples in the list belong to the same class. Once this happens, it simply creates a leaf node for the decision tree telling what class is chosen. It might also happen that none of the features give any information gain, so in this case C4.5 creates a decision node higher up the tree using the expected value of the class. It also might happen that no any instances of a class have ever been seen; again, C4.5 creates a decision node higher up the tree using expected value.

C4.5 is a landmark decision tree program that is very commonly used in practice. It has been demonstrated to perform as well as other state-of-the-art rule learners yet avoids their complex and ad hoc heuristics.

2.5 Tagger and Lemmatizer

To obtain the part of speech of words and count the lemmas in corpus, this thesis uses GPoSTTL tagger [33]. GPoSTTL tagger is an enhanced Parts-of-Speech Tagger using Brill's tagset, with a built-in tokenizer and lemmatizer. It is based on LPost package by Jimmy Lin. LPost itself is based on Benjamin Han's ePost package, which is a cleaned-up version of Eric Brill's original code. The primary lemma list of GPoSTTL tagger was compiled by Prof.

Yasumasa Someya.

GPoSTTL tagger is released under a GNU GPL compatible license. There still are some errors in lemmatized result, for example, *can't* is divided into '*ca*' and '*n't*'. Because it does not seriously affect our statistics of sentiment phrases, in later experiments we simply ignored this kind of error which results from tokenization, tagging or lemmatization.

The obvious benefits of GPoSTTL is that it integrates tokenizer, tagger and lemmatizer together; in addition, a potential advantage of GPoSTTL is that it converts all uppercase letters to lower case, preventing possible excessive overfitting which results from capital letters. This problem has been described in the paper by Andrew Lacey [20].

2.6 Link Grammar Parser

Sentiment Analysis needs not only unigrams of substantives, but also necessary bigrams that comprise modifiers and the adjectives or adverbs they modify. Substantives include adjectives, adverbs, nouns and verbs. Modifiers are negations, intensifiers, and diminishers that change the sentiment orientation of other terms. Our experiments use Link Grammar Parser [52] to capture bigrams.

The Link Grammar Parser is a syntactic parser of English. It is based on link grammar, an original theory of English syntax. Given a sentence, the system assigns to it a syntactic structure, which consists of a set of labeled links connecting pairs of words. The parser also produces a "constituent" representation of a sentence (showing noun phrases, verb phrases, etc.).

The parser has a dictionary of about 60000 word forms. It has coverage of a wide variety of syntactic constructions, including many rare and idiomatic ones. The parser is robust; it is able

to skip over portions of the sentence that it cannot understand, and assign some structure to the rest of the sentence. It is able to handle unknown vocabulary, and make intelligent guesses from context and spelling about the syntactic categories of unknown words. It has knowledge of capitalization, numerical expressions, and a variety of punctuation symbols.

Link Grammar Parser provides syntactical analysis at sentence level. The structure assigned to a sentence by a link grammar is rather unlike any other grammatical system that we know of (although it is related to dependency grammar). Rather than thinking in terms of syntactic functions (like subject or object) or constituents (like "verb phrase"), one must think in terms of relationships between pairs of words. In this thesis, we use the connection between words to extract the bigrams we need.

In the sentence shown in Figure 2.7, for example, there is an "EE" relation, which connects adverbs to other adverbs, between "pretty" and "well". Some adverbs can modify other adverbs ("very", "quite"); these carry EE+ connectors. EE can also be used with E ("He very quickly left"), CO ("Very quickly, he left") and EB ("He is very clearly a good programmer"). All these types of connections are the bigrams we want to extract, including valence shifters and original adjectives or adverbs.

```
++++Time                                          0.00 seconds (12.20 total)
Found 1 linkage (1 with no P.P. violations)
  Unique linkage. cost vector = (UNUSED=0 DIS=1 AND=0 LEN=12)

    +-------------------------Xp-------------------------+
    |      +-----Wd----+    +-----------Pv-----------+     |
    +--Wc--+X+    +-CO-+-Spx+       +---EE--+----E---+     |
    |      | |    |    |    |       |       |        |     |
LEFT-WALL so , then.e you are.v pretty.e well.e finished.v .

Constituent tree:

(S So ,
   (S (PP then)
      (S (NP you)
         (VP are
             (VP (ADVP pretty well)
                 finished))))
```

Fig 2.7 A parsed sentence by Link Grammar

There are 107 link types in Link Grammar as follows:

A	AA	AF	AL	AM	AN	AZ	B	BI	BT	BW	C	CC	CO	CP	CQ	CX	D	DD	DG	DP	DT	E	EA
EB	EC	EE	EF	EI	EL	EN	ER	EZ	FL	FM	G	GN	H	I	ID	IN	J	JG	JQ	JT	K	L	LE
LI	M	MFMG	MV	MX	N	ND	NF	NI	NJ	NN	NO	NR	NS	NT	NW	O	OD	OF	ON	OT	OX		
P		PF		PP		Q	QI		R	RS	RW	S	SF	SFI		SI		SX		SXI			
TA	TD	TH	TI	TM	TO	TQ	TS	TT	TW	TY	U	UN	V	W	WN	WR	X	Y	YP	YS	Z			

Fig 2.8 Link Types in Link Grammar Parser

In which we are looking for 'EE','E','CO','EB','EBm','EA','Os' structures of modified bigrams, and enrich them into the feature set.

Moreover, for successfully extracting efficient bigrams containing sentiment information, we follow P.D.Turney's patterns [14] which will be discussed in detail in section 5.1.3. To accurately capture those patterns derived from Turney's paper, we use the constituent tree,

which is under the linkage graph in Figure 2.7, to pick out the adjoining 'VP' and 'ADJP' or 'VP' and 'ADVP' structures. All of Turney's patterns are shown in Table 5.1.

2.7 WordNet::Similarity Package

2.7.1 General Introduction

WordNet is particularly well suited for similarity measures, since it organizes nouns and verbs into hierarchies of is–a relations. In WordNet version 2.1, there are nine separate noun hierarchies that include 80,000 concepts, and 554 verb hierarchies that are made up of 13,500 concepts.

WordNet::Similarity is a freely available software package that makes it possible to measure the semantic similarity and relatedness between a pair of concepts (or synsets). It provides six measures of similarity, and three measures of relatedness, all of which are based on the lexical database WordNet. These measures are implemented as Perl modules which take as input two concepts, and return a numeric value that represents the degree to which they are similar or related.

2.7.2 Six Similarity Measures and Three Relatedness Measures

WordNet::Similarity implements measures of similarity and relatedness that are all in some way based on the structure and content of WordNet. Measures of similarity use information found in an 'is-a' hierarchy of concepts (or synsets), and quantify how much concept A is like (or is similar to) concept B. The functions are called for a pair of word senses, and return the relatedness value For example, the relatedness of verb 'lose' and verb 'need' could be calculated by their overlaps in WordNet glosses as following:

```
lose#v#1 need#v#1 16 # verb lose versus verb need
```

Whereas, the relatedness between 'stupidity' and 'problem' is

```
stupidity#a#1 problem#a#1 32 # adj stupidity versus adj problem
```

In above two examples, the character between two pound symbols indicates the part of speech used for calculating the similarity between two words. For example, 'v' means verb, 'a' means adjective, 'n' means noun, and 'r' denotes adverb. Naturally, the first command requests the similarity for the first verb sense of 'lose' and the first verb sense of 'need'.

However, concepts can be related in many ways beyond being similar to each other. For example, a wheel is a part of a car, night is the opposite of day, snow is made up of water, a knife is used to cut bread, and so forth. As such WordNet provides relations beyond 'is-a', including has-part, is-made-of, and is-an-attribute-of. In addition, each concept is defined by a short gloss that may include an example usage. All of this information can be brought to bear in creating measures of relatedness. As a result these measures tend to be more flexible, and allow for relatedness values to be assigned across parts of speech (e.g., the verb murder and the noun gun). Therefore, we have the relatedness measures to judge the semantic relatedness between adjectives and adverbs.

Generally speaking, in WordNet, the similarity is measured between concepts in same part of speeches. There are six similarity measures. Three measures are based on path lengths between a pair of concepts: *lch* (Leacock and Chodorow, 1998), *wup* (Wu and Palmer, 1994), and *path*, while the other three measures of similarity are based on the *information content* of the least common subsumer (LCS) of concepts A and B. Information content is a measure of the specificity of a concept, and the LCS of concepts A and B is the most specific concept that is an ancestor of both A and B. These measures include *res* (Resnik, 1995), *lin* (Lin, 1998), and *jcn* (Jiang and Conrath [57], 1997).

Moreover as mentioned above, WordNet also provides relatedness measures between different parts of speeche. There are three measures of relatedness which are more general in that they can be made across part of speech boundaries, and they are not limited to 'is-a' relations. These three measures are: *hso* (Hirst and St-Onge, 1998), *lesk* (Banerjee and Pedersen, 2003), and *vector* (Patwardhan, 2003). Thus, we have a total of nine measures in which there are six that deal within the same part of speech and three that work between parts of speech.

'is-a' relations in WordNet do not cross part of speech boundaries, so similarity measures are limited to making judgments between noun pairs (e.g., cat and dog) and verb pairs (e.g., run and walk). While WordNet also includes adjectives and adverbs, these are not organized into 'is-a' hierarchies so similarity measures can not be applied

In these nine measures, the *jcn*, *lesk*, *hso* and *wup* algorithms will be used together to compute the relatedness similarity between pairs of words or phrases for the SO_WN algorithm in chapter 5.

2.7.3 Introduction of Measures Used in This Thesis

- Measuring Word Similarity using *jcn* method (*jcn* Algorithm)

jcn (Jiang and Conrath, 1997)[57] uses corpus data to populate classes (synsets) in the WordNet hierarchy with frequency counts. Each synset is incremented with the frequency counts from the corpus of all words belonging to that synset, directly or via the hyponymy relation. The frequency data is used to calculate the Information Content (IC) of a class $IC(s) = -log(p(s))$. Jiang and Conrath specify a distance measure: $D_{jcn}(s_1,s_2)=IC(s_1)+IC(s_2)-2IC(s_3)$, where the third class (s_3) is the most informative, or most specific, superordinate synset of the two senses s_1 and s_2 This is transformed from a distance measure in the WN-Similarity package by taking the reciprocal:

$$jcn(s_1,s_2)=1/D_{jcn}(s_1,s_2)$$

Measures of relatedness are more general in that they can be made across part of speech boundaries, and they are not limited to 'is-a' relations. As montioned above, there are three such measures in the package: *hso*, *lesk*, and *vector*, in which *lesk* (Banerjee and Pedersen, 2003) [58] will be used to calculate the similarity between adjective pairs and adverb pairs. In addition, *lesk* is also used with *jcn* measure together for computing similarity of noun pairs and verb pairs. All detail of SO_WN algorithm is described in chapter 5.

- Measuring Word Similarity using Lesk's method (*lesk* Algorithm)

In WordNet::Similarity package, *lesk* (Banerjee and Pedersen, 2002)[58] score maximizes the number of overlapping words in the gloss, or definition, of the senses. It uses the glosses of semantically related (according to Word-Net) senses.

The word similarity/relatedness study originated from the requirement of word disambiguation. Lesk (1986) [31] initiated a simple idea that a word's lexical definitions are likely to be good indicators for the senses they define. For example, there are two distinct word senses of *bank* are as following:

1. Some people on the bank called out to the man in the *boat*...
2. He wants to get back funds on *deposit* with a broker of TD *bank*.

If either *boat* or *deposit* occurs in the same context as *bank*, then people tend to assume that the occurrence belongs to the meaning whose definition includes that word: sense 1 for *boat*, sense 2 for *deposit*.

Therefore, Lesk [31] assumed that $D_1 \ldots D_K$ to be the dictionary definitions of the senses $S_1 \ldots S_K$ of the ambiguous word w, represented as the bag of words occurring in the lexical definition, and E_{vj} the lexical definition of a word V_j appearing in the context c of w, represented as the bag of words appearing in the definition of V_j. (Here, $S_{j1} \ldots, S_{jL}$, are the senses of V_j, then E_{vj}

$=\bigcup_{j_i} D_{j_i}$. The sense differences for the words V_j that occur in the context of w are ignored.)

Thus, Lesk's algorithm could be summarized as:

$$score(s_k) = overlap(D_k, \bigcup_{v_j inc} E_{v_j})$$

The overlap function means counting the number of common terms. Finally, for all senses S_K of word w, after k iterations, select the sense S_K with a maximum $score(s_k)$ as the disambiguated word sense.

- Computes the Semantic Relatedness of Word Senses by Hirst and St-Onge [37] (*hso* algorithm)

Hirst and St-Onge [37] issued a method which is an edge-based method that excludes some specific types of edges. This method could be used to build lexical chains. They measure the semantic relatedness of words in text could be used to identify the links of the lexical chains. Their original goal is to recognize malapropisms (a kind of spelling errors). Their method is based on two hypotheses: First, a malapropism word is not likely to be inserted in any chain with other words, so words that cannot be inserted with other words can be considered as malapropisms; Second, if a spelling replacement can be found and can be inserted in a chain with other words, this replacement is likely to be the intended word for which a malapropism has been substituted.

The algorithm proposed in WordNet::Similarity package uses WordNet to automatically quantify semantic relations between words. Because this algorithm actually spans the gap between synsets, it could be used between different parts of speech. Therefore, we can use its flexibility to calculate the relatedness between words without regarding to their POS.

- Measuring Word Similarity based on the path lengths (*wup* algorithm)

The methods in WordNet Package can be classified into two categories, edge-based methods and node-based (information content-based) methods. Edge based methods attempt to

measure the distance between two senses according to the length of the path between them in the semantic networks. The simplest method is to count the number of edges or nodes between them. Node-based methods measure the distance between two senses according to the statistical information contained in the nodes within the semantic network. Obviously, the edge based methods are more appropriate to our scoring task because it is more straightforward to present the distance between a pair of sentiment words.

There are three edge based similarity measures: *lch* (Leacock and Chodorow, 1998), *wup* (Wu and Palmer [66], 1994), and *path*. These three methods are based on path lengths between pairs of concepts. *wup* finds the depth of the LCS of the concepts, and then scales that by the sum of the depths of the individual concepts. The depth of a concept is simply its distance to the root node. Due to the good performance of the *wup* method, we choose it as one of the measures used to calculate the relatedness between sentiment terms.

2.8 General Inquirer

In this thesis, positive and negative terms are initially taken from the *General Inquirer* (Stone et al., 1966) (hereafter GI). GI is a dictionary that contains information about English word senses, including tags that label them as positive, negative, negation, overstatement, or understatement. General Inquirer plays an important role in the feature selection process of our experiments.

2.9 Stop Word List

Stops words, such as conjunctions and prepositions, are considered to provide no information gain. It is a widely accepted technique to remove these words from a corpus to avoid the noise and reduce feature set size. The pre-processing of the raw data is done in Perl. For eliminating stop words, we combined two resources from [35] and [36].

We extend the stop word list by including some domain-specific high-frequency uninformative words for bank reviews such as bank names etc. Those words do not help our learning or unsupervised scoring process, but cause overfitting.

Chapter Three

Preliminary Study

In this chapter, we will review previous works of sentiment analysis, discuss their advantages and limitations, and present our speculation, assumptions and hypotheses. Although almost all previous studies focus on binary sentiment classification rather than multiclass prediction, their approaches inspire us with heuristics, provide us with suggestions, and guide the formation of our methodology.

3.1 Research based on knowledge-based or human-structured methods

3.1.1 The Origin of Sentiment Analysis-Adjective Orientation

In general, sentiment analysis (hereafter SA) research can be traced back to the work by V. Hatzivassiloglou and K. R. McKeown (actually many other research, such as the work of R.Passonneau and D. Litman. 1993 [24], focused on sentiment analysis too, but these tasks paid a little attention to the correlation between linguistic clues and semantic sentiments, so generally is not commonly accepted as the origin of SA). Afterwards, the high accuracy of their prediction inspired subsequent works and made SA a promising area.

Bruce and Wiebe (2000) [53] performed a statistical analysis of the assigned classifications, finding that adjectives are statistically significantly and positively correlated with subjective sentences in the corpus on the basis of the log-likelihood ratio test statistics G^2. Thus, for sentiment analysis, adjectives are the most important type of part of speech (hereafter POS), because they carry more semantic orientation information than other types of POS. J.M.Wiebe's work [22,23] had already proved the high correlation between the subjective

favorability and the presence (attention: not frequency) of adjectives. Moreover, Hatzivassiloglou and K. R. McKeown 1997 [13] used unsupervised learning methods in semantic orientation prediction based on 21 million words from the 1987 Wall Street Journal corpus, and reported 92% accuracy on the classification task on 1336 adjectives. This accuracy is almost the highest reported accuracy of sentiment classification for words. In other words, Hatzivassiloglou's experiments once again presented the importance of adjectives. Although this work was about classifying the orientation of adjectives and not of sentences or documents, their special emphasis on the adjectives reminds us to pay more attention to this POS.

Hatzivassiloglou's method relies on an observation that, in conjoined adjectives, there are always linguistic constraints imposed on the semantic orientations. For example, *and* generally connects a pair of synonyms or adjectives with same orientation, while *but* usually conjoins antonyms. In other words, a connective and its arguments are mutually constrained as described by the sentence [13]:

$$\text{The tax proposal was} \begin{bmatrix} simple & and & well-received \\ simplistic & but & well-received \\ *\,simplistic & and & well-received \end{bmatrix} \text{by the public.}$$

Obviously, the third conjunction with '*' is wrong due to breaking the linguistic constraint.

Hatzivassiloglou's work identified and classified sentiment-oriented adjectives in the following 4 steps:

1. All conjunctions of adjectives are selected from the documents along with their morphological information.
2. A log-linear regression approach combines 3 morphological attributes to label these conjunctions according to whether each pair of conjoined adjectives are of the same or different orientation. The 3 morphological attributes are:
    i. the conjunction used (and, or, but, either-or, or neither-nor),
    ii. the type of modification (attributive, predicative, appositive, resultative)

iii. the number of the modified noun (singular or plural)

This step holds an 82.05% accuracy.

Table 3.1 shows the result of his approach:

Prediction method	Morphology used?	Accuracy on reported same-orientation links	Accuracy on reported different-orientation links	Overall accuracy
Always predict same orientation	No	77.84%	-	77.84%
	Yes	78.18%	97.06%	78.86%
But rule	No	81.81%	69.16%	80.82%
	Yes	82.20%	78.16%	81.75%
Log-linear model	No	81.53%	73.70%	80.97%
	Yes	82.00%	82.44%	**82.05%**

Table 3.1 Accuracy of several link prediction models [13]

3. Place as many words of the same orientation as possible into the same subset by clustering adjectives into two subsets of different orientation.
4. The subset with the highest frequency is labeled as positive adjectives, which obtains 92.37% accuracy. (In reality, most adjectives of conjunctions have positive orientation).

The overall results of this step are shown in Table 3.2:

α	Number of adjectives in test set ($\mid A_\alpha \mid$)	Number of links in test set ($\mid L_\alpha \mid$)	Average number oflinksfor each adjective	Accuracy	Ratio of average group frequencies
2	730	2,568	7.04	78.08%	1.8699
3	516	2,159	8.37	82.56%	1.9235
4	369	1,742	9.44	87.26%	1.3486
5	236	1,238	10.49	**92.37%**	1.4040

Table 3.2 Evaluation of the adjective classification and labeling methods [13]

It is worth mentioning that Hatzivassiloglou's experiment has 3 remarkable important hypotheses:

1. Guessing that a conjunction is of the same orientation type achieves high level of performance.
2. *but* is one exception since it connects antonyms, so guessing that the two adjectives

conjoined by *but* have different orientation could increase the accuracy.

3. Morphologically related adjectives (e.g. *adequate-inadequate* or *thoughtful-thoughtless)* almost always have different semantic orientations.

The third conclusion is very useful for this thesis to exclude morphologically related adjectives from features when using WordNet to measure the relatedness between adjectives and adverbs.

3.1.2 Sentence Level Subjectivity

Once word level orientation is determined, sentence level subjectivity might be the next step. Based on Hatzivassiloglou's work at adjectives [13], Hatzivassiloglou and J.M.Wiebe [16] investigated whether certain lexical features of adjectives benefit the prediction of sentence-level subjectivity. They were looking for an approach to split subjective sentences from objective sentences. They put dynamic adjectives, semantically oriented adjectives, and gradable adjectives into a simple subjectivity classifier, and concluded that these adjectives are strong predictors of subjectivity. (J.M.Wiebe et al. explored a system that identifies opinionated sentences [25] and executes subjectivity tagging [26], whereas their experimental corpus was somewhat specific and *recall* is relatively low. However, their result is still often used as baseline accuracy in subsequent recognition systems.).

This sentence level subjectivity study relied on two hypotheses: First, it had already been proven in Hatzivassiloglou and K. R. McKeown 1997 [13] that the orientation of adjectives is a comparatively objective semantic property, because they obtained 92.37% accuracy in polarity assignment for adjectives. In other words, as a syntactical functional attribute, orientation can be used to make the modified items better (positive) or worse (negative), so it is an ideal measure to sentiment analysis. Second, the gradability of adjective is another good predictor of subjectivity. Generally, there are two types of grading modifiers:

1. Inflected forms of adjectives, such as good-better-best;

2. Adverb modifiers, such as very;

According to previous linguistic studies, gradability has the ability to intensify or diminish the modified noun. In addition, the comparative and superlative of adjectives and adverbs are in common use in customer reviews.), so it is not hard to extract them from the corpus. Subsequently, the semantic orientation and gradability of adjectives are used as important extensions to features used for classifying sentences by subjectivity.

When the algorithm finds either one of the above types of grading modifiers, an adjective is automatically labeled as gradable; otherwise, if no grading modifiers have been found, the algorithm uses a log-linear model to count the number of times an adjective has been found in gradable context and the number of times it has been observed in a non-gradable context, then calculates the gradability with these two numbers.

Hatzivassiloglou and J.M.Wiebe's experiment adopted different combinations of four kinds of features:

1. All adjectives;
2. Dynamic adjectives which are indicative of subjective sentences;
3. Orientation labels assigned by Hatzivassiloglou and K. R. McKeown 1997 [13];
4. Other gradibility labels generated by log-linear model

Finally, their experiment reached the conclusion that using automatically calculated gradibility features with polarity features (such as the presence of key adjectives) together is better than using polarity features only or using polarity features plus manually selected gradibility features. Recognizing subjectivity from a sentence might benefit document level polarity assignment.

This conclusion noticed that the modifiers including intensifiers, diminishers, and negations might also benefit the feature selection as the gradibility features do.

3.1.3 SO-PMI-IR Method (Semantic Orientation of Pointwise Mutual Information by Information Retrieval)

The SO-PMI-IR method was first introduced by Peter D.Turney [14]. Due to its stability, consistency and flexibility, various PMI-IR approaches have been derived and extensively adopted. Using Pointwise Mutual Information (hereafter PMI) allows sentiment analysis to take advantage of supervised machine learning methods. To some extent, SO-PMI-IR is a milestone of recent sentiment analysis studies.

In SO-PMI calculation, the semantic orientation of a phrase is calculated as the mutual information between the given phrase and the word "excellent" minus the mutual information between the given phrase and the word "poor". Finally, a review is labeled as recommended if the average semantic orientation of its phrases is positive, or else, negative.

Mutual information is a symmetric, non-negative measure of the common information in the two variables. It describes the independence between two variables X and Y [2]:

$$I(X;Y) = H(X) - H(X \mid Y);$$

- It is 0 only if two variables are totally independent;
- For two dependent variables, mutual information grows not only with the degree of dependence, but also according to the entropy of the variables.[2]

When talking about pointwise mutual information (PMI), we mean the mutual information between two particular points as defined in the following expression (Church and Hanks 1989 [48]):

$$I(x,y) = \log \frac{p(x,y)}{p(x)p(y)}$$

The PMI is roughly a measure of how much information one word gives about the other, so it can be used when measuring the polarity of a word.

Under some complex or exceptional situations, Hatzivassiloglou's solution might incorrectly

classify phrases/sentences, but SO-PMI method is more reasonable and more reliable than the former methods relying on a single adjective. For example, "unpredictable plot" implies a positive opinion in movie reviews, although 'unpredictable' is negative in general [14]. However, Turney's method treats phrase as a smallest unit to calculate Semantic Orientation (SO) as following [14]:

$$SO(phrase) = PMI(phrase, "Excellent") - PMI(phrase, "poor")$$

Based on the function provided by AltaVista (no longer available), via mathematical manipulation, and using NEAR function, the SO was elaborated as [14]:

$$SO(phrase) = \log_2 \left[\frac{hits(phrase\ NEAR\ "excellent")\ hits("poor")}{hits(phrase\ NEAR\ "poor")\ hits("excellent")} \right]$$

Because this method involves more semantic and contextual syntactical factors, it is more feasible than Hatzivassiloglou's method.

Finally, the review is labeled as positive if the average SO of all phrases is positive, or else negative if the average SO is negative.

Turney's SO-PMI-IR method is executed on different domains from Epinions [27] including 'Automobiles', 'Banks', 'Movies' and 'Travel Destinations'. Interestingly, the experimental result presents relatively large difference between domains. The result showed as follows:

Domain of Review	Accuracy	Correlation
Automobiles	**84.00 %**	0.4618
Banks	80.00 %	**0.6167**
Movies	**65.83 %**	0.3608
Travel Destinations	70.53 %	0.4155
All	74.39 %	0.5174

Table 3.3 The accuracy of the classification and the correlation of the semantic orientation with the star rating [14]

As shown in Table 3.3, SO-PMI-IR archieve the highest accuracy of 84.00% on reviews of automobiles and the lowest accuracy of 65.83% on reviews of movies. Regarding the lower accuracy on movie reviews, Turney gave two reasons in his explanation:

1. Movie reviews tend to use some negative words in positive reviews, and vice versa.

2. Movie reviews somehow ignore the fact that some phrases actually discuss other objects, such as actors and events, instead of the movie itself.

The first reason reveals one of the most challenging factors of sentiment analysis: that opinions are usually hidden in a concealed and periphrastic expression, and need to be dealt with using knowledgeable semantic and linguistic tools. For example, the "thwarted expectations" narrative mentioned in Bo Pang and Lillian Lee et al. [10] is a very implicit way for expressing authors' subjectivety. Without complex and sophisticated preprocessing, it is difficult to recognize the real sentiment orientation of such kind of sentences.

The second reason involves some hot spots of text-categorization, such as entity recognition and entity-sentiment association. This area of studies has already shown progress, such as the work by J.Yi, T.Nasukawa, R.Bunescu and W.Niblack [12] to be discussed shortly.

Due to its challenging character, as a result, movie reviews have been a common testbed in many subsequent researche on sentiment analysis.

The result of Turney's research presents three valuable suggestions to us:

First, SO is highly domain-specific. Therefore, the feature selection has to be against each domain respectively; whereas an open domain feature set is too infeasible to obtain good performance for sentiment analysis.

Second, the reason why SO-PMI-IR measure results in good performance is that it utilizes

semantic, contextual, and syntactical information better than previous approaches to sentiment analysis. However, is SO-PMI-IR the best way to detect the potential sentiment orientation of terms? In fact, WordNet is an experienced and practiced resource provider which is enriched with hierarchical structural semantic and syntactical information of English words. Therefore, incoporating the precise and abundant external relational information from WordNet measures and contextual information into SO-PMI-IR algorithm may provide more effective evidence for accurately judging SO of terms than SO-PMI-IR alone does. In chapter 4, we will investigate the limitation of computing SO-PMI-IR method by sending queries to AltaVista, and discuss the possibility of using WordNet measures to calculate SO. Moreover, in chapter 5, we will explore the effect of WordNet measures in feature selection, and report the comparison results of using presence, frequency, and WordNet score SO_WN as the feature for learning.

Lastly, line 2 of Table 3.3 shows that the semantic orientation and five star rating correlate on bank reviews much more than on other datasets, which is the reason for using bank reviews as the dataset in this thesis. The highest correlation value 0.6167 indicates that semantic orientation influences bank reviews more than other reviews on the score of five star rating. Intuitively, we believe that higher correlation value will bring more positive effects to the classification result.

Therefore, when we transform the five star rating into a multiclass classification problem in chapter 5, the learning result will be more typical, more clear, and more representative than other types of reviews (such as in Table 3.3, the domain of 'Automobiles', 'Movies' and 'Travel Destinations') for observing the effect of learning algorithms and judging whether subsequent improvement approaches work.

3.1.4 (Subject, Sentiment) Association by Relationship Analysis

There is an important problem regarding whether we need to distinguish between different topics and entities first in our corpora before scoring the reviews of subjects.

In the work of J.Yi, T.Nasukawa, R.Bunescu and W.Niblack [12], they raised a problem about topic association. They found that there are two challenging aspects of sentiment analysis: First, although the overall opinion about a topic is useful, it is only a part of the information of interest. Document level sentiment classification fails to detect sentiment about individual aspects of the topic. Second, the association of the extracted sentiment to a specific topic is difficult, so there is a difficulty to ensure all of the extracted sentiment features is about the topic of review; in other words, it is possible that some comments about independent entities affect the sentiment classification of the whole review. For example, a customer may be happy about his camera, but dissatisfied by the included memory card .

Generally, statistical opinion extraction algorithms deal with the above problem by either: 1) assuming the topic of the document is known a priori or 2) simply associating the opinion to a topic term co-existing in the same context.

J.Yi, T.Nasukawa, R.Bunescu and W.Niblack [12] contribute a sentiment analyzer that not only evaluates sentiment but also detects all references to the given subject, thereby assigning the sentiment to topical references instead of to the whole document. Their sentiment analyzer includes three main parts: 1) a topic specific feature term extraction, 2) sentiment extraction, and 3) (*subject, sentiment*) association by relationship analysis.

Consequently, we need to determine whether we need to implement a similar analyzer to identify different entities and topics from the reviews of bank; in other words, whether we need to distinguish the opinions of bank services and bank products from the opinions of banks.

Therefore, we proposed a test: using the Link Grammar Parser to build a simple lexicon as:

```
<subject> <adjective> <sentiment orientation>
<subject> <verb> <sentiment orientation>
```

The ' <subject> <adjective>' or ' <subject> <verb>' are extracted from the sentences of each review which is analyzed by Link Grammar Parser; while the sentiment orientation is obtained from General Inquirer dictionary. This lexicon focuses on the bigrams comprising a subject plus an adjective or a predicate verb; then we check the consistency of the sentiment orientation between services, products and the bank in each review.

If the subject is the name of the bank in question or simply a word 'bank', we consider this bigram to be of bank; else, we classify them into the bigrams for services or products. No pronouns are included, because analyzing the substitutional relationship between a pronoun and its subject is out of the scope of this thesis.

	Opinions about the bank	Opinions about services or products	Agreed opinions	Percentage in agreement
<subject> <adjective>	2936	3821	2525	**86%**
<subject> <verb>	2560	3730	2022	**79%**

Table 3.4 Agreement of the opinions between services, products and the bank

Table 3.4 shows the number of bigrams extracted from all 3164 bank reviews, and the 'percentage in agreement' between the opinions of services/products and the opinions of banks. The percentage is computed by dividing the number of consistent opinions by the total number of opinions of banks.

From Table 3.4, we can conclude that the opinions of banks and the opinions services are positively correlated, and their sentiment orientation is basically consistent. Therefore, we do not need to distinguish the opinions of services/products and the opinions of banks. We assume that for each customer review, its sentiment orientation is consistent with that of its services and bank products.

3.1.5 Refining the identification of sentiment vocabulary (+improved sentence level subjectivity)

Michael Gamon and Anthony Aue [17] raised their own assumption that sentiment terms of opposite orientation tend not to co-occur at the sentence level. Because they considered the document level too coarse, they modified Turney's method [14] for feature selection and extended the set of labels by classifying sentences into 'positive', 'negative', and 'neutral' classes.

Michael Gamon et al. [17] constructed their own SM+SO method to select features. In this method, the SO is the sentiment orientation formula provided by Turney [14], and SM is so-called 'sentiment mining' (hereafter SM) method. For the SM method, they selected terms that have the lowest PMI scores on the sentence level with respect to a set of manually selected seed words. According to their assumption, terms should have low association at the sentence level, so the low-PMI-scoring terms will be particularly rich in sentiment terms. In a result, Michael Gamon's assumption allowed them to identify sentiment terms reliably, and then they used these terms for classifying sentences by their sentiment orientation.

	Avg precision	Avg recall
SO	0,4481	0.4511
SM + SO	0.4568	0.4605
SM + SO multi-iteration	0.4957	0.4995
SM + SO multi-iteration + NB Bootstrap	0.5167	0.52

Table 3.5 Results obtained by SO, SM+SO, SM+SO multi-iteration and SM+SO multi-iteration + NB Bootstrap algorithms[17]

Table 3.5 shows the classification results by four different algorithms. There is little improvement between SO, SM+SO algorithms. Although the precision did not increase apparently, SM successfully reduced the numbers of features from 13,000 to 2,600.

In line 3 of Table 3.5, we see that both precision and recall increased by several percentage

points. This improvement is resulted from the increased number of seed features through the SM feature selection method. Based on the basic SO + SM method, they used multiple iterations to gradually build the list of seed words. During this process, the number of seed features increased from 10 to 111, this augmentation significantly improved the accuracy of prediction.

Finally, the line 4 of Table 3.5 shows that bootstrapping brought an obvious improvement to the Naïve Bayes algorithm. They used the SO classifier to label a subset of data, and then used this data to bootstrap the Naïve Bayes classifier.

There are four aspects worth noting:

First, combining the SM and SO methods not only dramatically reduced the required number of features, but also outperformed using the SO method alone. This result suggests that SO alone is not a very good measure for sentiment feature selection, and there is a lot of room for improvement. To some extent, SM brought more experienced semantic sentiment information into the feature selection process, so incorporating SM with SO obtained a better effect than using SO alone. Therefore, we use WordNet as a secondary important reference for measuring the SO of terms.

However, secondly, there is a fatal weakness in Michael Gamon and Anthony Aue's method: they used manually selected in-domain seed words to calculate PMI. This idea is in conflict with Bo Pang's [10] work which showed that humans' intuition may not work well for choosing discriminating sentiment words. Actually, this could be the reason why Michael's experiment did not achieve a satisfactory accuracy for predicting the SO of sentences. Consequently, when computing the SO-PMI-IR and SO_WN score in chapter 5, we will choose the seed words automatically.

Thirdly, bootstrapping contributed much to improving the performance of classification by Naïve Bayes. However, the bootstrapping procedure may produce a good result for very small dataset, but does not necessarily benefit our learning result on the dataset of 3164 reviews. In

addition, bootstrapping often produces unreasonable error prediction when the distribution of classes is especially abnormal. The classes of bank reviews are unevenly distributed among five classes, so bootstrapping is not appropriate for exceptional minority classes. Therefore, we will look for other approaches and will not use bootstrapping in our experiments.

Finally, Michael Gamon et al. adopt multiple iteration method to gradually build new SM seed words. This approach took effect in each round of iterations, and produces remarkable improvement. This may be a promising procedure when we generate seed words for calculating SO_WN score with WordNet.

Moreover, Michael Gamon and Anthony Aue believed that unsupervised learning (or some weakly supervised methods) is feasible because it avoids building a new set of training data when switching to a new domain. In fact, their opinion is consistent with P.D.Turney's [14] conclusion. Subsequently, we also look at unsupervised learning, and discuss its performance, in chapter 5, compared to the results of supervised learning.

3.1.6 A straightforward quantitative sentiment scoring

Andrew Lacey [20] considered the sentiment of documents to be essentially a continuous spectrum. According to Lacey's idea, ranking documents in order of favorability instead of classifying them into two orientations is a more reasonable solution.

This solution could be divided into two parts:

a) The first part is generating a lexicon of sentiment elements. The sentiment elements consist of a list of unigrams, a list of phrases or a list of linguistic structures.
b) The second part is assigning a sentiment value to each document in corpus.

From Lacey's point of view, the sentiment data, as a subjective measure, should vary for

different domains. He suggested that sentiment analysis should use manually developed domain-oriented lexicon rather than a cross-domain common list of sentiment dictionary.

In the first part of Lacey's algorithm, he counted each word only once per training document and used the information in a lexicon to calculate the average scores for each word as the following formula [20]:

$$S(w) = \frac{\sum[S(d) \mid w \in d]}{|(D : d \mid w \in d)|} \quad \textbf{(3.1)}$$

In this formula, d represents a document, w represents a word, D represents a set of documents, and $S(w)$ represents the score of item w.

Then, in the second part of assigning scores to documents, an inverse formula is implemented. Each test document is assigned the score equal to the average of the scores of the words appearing in the document. All the words in the document but not in the list of sentiment words (the lexicon) are ignored. The formula is like follows [20]:

$$S(d) = \frac{\sum[S(w) \mid (w \in d) \wedge (w \in L)]}{|[W : (w \in d) \wedge (w \in L)]|} \quad \textbf{(3.2)}$$

In this formula, L is the list of words resulting from the term-extraction process and W is a set of words.

Lacey's goal was to output a ranking of documents. In his experiments, the above algorithm was excuted on two different datasets. The results on both datasets are highly consistent. The results of the automatic ranking are in accordance with the results from human-assigned ranking.

Of course, the ranking result is not identical with human-assigned ranking, there were some exceptional points distributed in the whole range of all documents. Although Lacey's method

is rather coarse and cursory, it is a straightforward idea. Our unsupervised learning algorithm is derived from Lacey's approach and used as the baseline experiment. The unsupervised learning results on bank reviews will be discussed in Chapter 4 and Chapter 5.

3.2 Research based on machine learning methods

3.2.1 Binary sentiment classification using machine learning techniques

Some research gradually changed from knowledge-based semantic orientation analysis to machine learning-based sentiment analysis. Bo Pang, Lillian Lee and Shivakumar Vaithyanathan [10] presented their study of sentiment classification using machine learning techniques, and analyzed the reasons why sentiment analysis is more challenging than text categorization.

Contrary to intuition, Bo Pang et al. found that humans may not always obtain the best performance for choosing discriminating words to analyze sentiment orientation. For exploring whether manually selected sentiment words or automatically extracted ones more effective to classify opinions, they asked two graduate students to choose seven positive and seven negative sentiment indicator words from movie reviews, and then using a simple statistics function they created a list of seven positive and seven negative words. Finally, they observed that automatically generated features outperformed human-produced ones in sentiment classification, and concluded that using corpus-based feature selection technique is better than relying on intuition.

Bo Pang's investigation indicates that some traditional manually or semi-manually generated lexicons, which were used in previous sentiment analysis works by Huettner and Subasic[54], by Das and Chen[55], and by Tong [41], are actually questionable. To be fair, Pang did not exactly go about choosing the sentiment words in the most thorough way.

Afterwards, Bo Pang et al. [10] implemented machine learning algorithms, including Naïve Bayes (NB), Maximum Entropy (ME) and Support Vector Machines (SVM) on different feature sets. The features used in their paper comprise unigrams, bigrams, part of speech (hereafter POS), adjectives, and word position in a text etc.

	Features	# of features	Frequency or presence?	NB	ME	SVM
(1)	unigrams	16165	freq.	**78.7**	N/A	72.8
(2)	unigrams	16165	pres.	81.0	80.4	**82.9**
(3)	unigrams+bigrams	32330	pres.	80.6	80.8	**82.7**
(4)	bigrams	16165	pres.	77.3	**77.4**	77.1
(5)	unigrams+POS	16695	pres.	81.5	80.4	**81.9**
(6)	adjectives	2633	pres.	77.0	**77.7**	75.1
(7)	top 2633 unigrams	2633	pres.	80.3	81.0	**81.4**
(8)	unigrams+position	22430	pres.	81.0	80.1	**81.6**

Table 3.6 Average three-fold cross-validation accuracies, in percent [10]

As shown in Table 3.6, this study combined different types of features and generated eight different feature sets. Boldface indicates the best performance for a given feature set. There are several noteworthy results in Table 3.6:

1. Compared to a same number of unigrams (line 7), adjectives alone (line 6) provide less useful information and cause relatively poor performance.
2. The highest accuracy of classification is yielded by SVM with unigrams only (line 2).
3. Unigram presence beats unigram frequency and produces significantly better performance. (line 2 and line 1)
4. Incorporating bigrams into the same amount of unigrams (line 3) is outperformed by using that amount of unigrams alone (line 2); Moreover, the learning accuracy by bigrams (line 4) is much lower than that by a same number of unigrams (line 2).

The first result implies that although intuitively adjectives should be the richest POS of sentiment meanings, they are surpassed by the top most frequent unigrams. Our understanding of this result is that the importance of adjectives is counteracted by their relative infrequency, so 2633 adjectives do not necessarily outperform 2633 most frequent unigrams. This result suggests that frequency is a valuable factor that needs to be paid more attention to than the POS

of adjective because the former benefits classifiers more than the latter does.

The second point suggested by these results is that SVM is a promising classifier in sentiment classification. In Bo Pang's investigation, on five out of eight feature sets, SVM achieved the highest accuracy. Therefore, we select SVM as one classifier in our experiments of supervised learning.

Bo Pang's explanation of the third conclusion is that "the frequency of content words usually implies the degree a documents belonging to a topic, but *does not intensify the favorability*". We think that this explanation is debatable. Obviously, when a sentiment word is repeated many times in a review, the author's sentiment orientation about the subject s/he discusses is emphasized although the degree of author's favorability might not be necessarily directly proportional to the times the sentiment word appears. Therefore, the italic part in Bo Pang's explanation may not hold. The repetition of sentiment terms actually emphasizes the influence of these terms, so in this thesis we think of how to use the frequency of sentiment terms to weight their sentiment orientation. . Especially for multiclass classification, we speculate that the term frequency is highly correlated with the five star rating. Moreover, because we will utilize the order information of five star rating method in our study of supervised learning (in chapter 5), we expect that frequency value may benefit the ordinal meta-learning. Consequently, we will treat the presence, frequency, tf/idf and SO score of sentiment terms as different weighting approaches, explore their effect, and compare their performance in chapter 5. In a word, our assumption is that treating frequency as a weighting method is a more reasonable understanding of frequency than considering it useless; In addition, we speculate that frequency is a sentiment indicator either for text categorization or for sentiment analysis.

The fourth result implied that using mixed bigrams and unigrams are outperformed by using unigrams alone when Bo Pang et al experimented with Naïve Bayes and SVM classifiers. This result is instructive because we also need to pay more attention to the feature selection due to its crucial influence on the learning result. Interesingly, Bo Pang's observation is in opposition to the conclusion of contextual valence shifters research by Alistair Kennedy and Diana Inkpen

[11]. We will report our result observed in supervised learning, and reason out the underlying fact in chapter 5.

Furthermore, in Bo Pang's study, the Naïve Bayes presented comparable performance to that of sophisticated algorithms such as SVM. Natually, we also choose this simple but effective classifier in our supervised learning.

3.2.2 Sentiment Analysis using support vector machines

Bo Pang et al. [10] suggested that adding identification of features to indicate whether sentences are on-topic benefits the sentiment classification task. Afterwards, Tony Mullen and Nigel Collier [18] inherited Bo Pang et al's [10] hypothesis. They built hybrid SVM classifiers that incoprated topic information into unigram-type feature-based SVMs, and produce the best results on the movie review data. Mullen et al. [18] involved two new types of measures in their feature sets:

1. Topic Proximity features

   According to Mullen's assumption, although in opinion-based texts there is generally a single primary subject about whether the opinion is favorable or unfavorable, it would seem that secondary subjects may also be useful if identified. Therefore, they tried to find the second related subject from their dataset. In their experiments, texts were annotated by hand using the Open Ontology Forge annotation tool, as described in Tony Mullen and Nigel Collier [18]. Their dataset consisted of a total of 1380 imdb.com movie reviews, approximately half positive and half negative. In each instance of the dataset, in addition to the SO-PMI features, they added Osgood semantic differentiation values representing various relationships between topic entities and value phrases. In each record review, references (including co-reference) to the record being reviewed were tagged as THIS_WORK and references to the artist under review were tagged as THIS_ARTIST. Of

course, by their assumption, THIS_WORK and THIS_ARTIST are tightly correlated.

With these tagged entities, a number of new features may be extracted. These features represented various relationships between topic entities and phrases, such as THIS_WORK and THIS_ARTIST described above.

Mullen's method produced good performance for their SVM classifier. However, we don't identify the 'entiry <->sentiment phrase' relationships, because bank reviews focus on the service of a specific bank and generally has same orientation with the phrases for products. On the other hand, bank reviews differ from movie reviews because bank reviews do not have a lot of thwarted-expectations that use implicit expression to emphasize the opinion in contrast. Therefore, we do not need to recognize and identify 'entiry<->sentiment phrase' relationships for our bank reviews.

2. Syntactic-relation features

In Mullen et al's paper [18], they derived other feature types using the method of Kamps and Marx [64] which uses WordNet relationship to derive three values pertinent to the emotive meaning of adjectives. The three values are *potency* (strong or weak), *activity* (active or passive) and the *evaluative* (good or bad) introduced in Charles Osgood's Theory of Semantic Differentiation [65].

Because these three Osgood values depend on the synonymy synset of WordNet and only adjective and adverb are organized in synonym synset, Mullen et al. build the three values EVA(*evaluative*), POT(*potency*), and ACT(*activity*) on a list of 5410 adjectives.

The results of Mullen's experiments are shown in Table 3.7:

Model	3 folds	5 folds	10 folds	20 folds	100 folds
⋮	⋮	⋮	⋮	⋮	⋮
Turney Values only	68.4%	72%	68.3%		
Uniframs	82.8%	79%	83.5%		

Uniframs and Turney	83.2%	84%	85.1%		
Hybrid SVM(Turney and Lemmas)	84.4%		**86.0%**		
Hybrid SVM(Turney/Osgood and Lemma)	**84.6%**		**86.0%**		
Lemmas	84.1%	83%	85%	**85%**	86%
Lemmas and Turney	84.2%	84%	85%	**85%**	86%
Lemmas and PMI		84%	85%	**85%**	86%
Hybrid SVM(PMI/Osgood and Lemmas)		**86%**	**87%**	84%	**89%**

Table 3.7 Accuracy results for 3, 5, 10, 20 and 100-fold cross-validation tests on IMDB movie reviews [18]

According to the Table 3.7, the results of line 6 and line 10 in which the Osgood's value were used achieve the highest accuracy of sentiment classification. Obviously, this result suggests that topical and synthetic information indeed benefit the learning process, so relying on WordNet we proposed our own synthetic features Average_PMI, Average_lesk, Average_hso, Average_jcn in our experiments, and explore their effects to the learning results. We will discuss them in detail in chapter 4 and chapter 5.

In Mullen's [18] research, another valuable conclusion worth noting is that the accuracies based on lemmas outperformed that based on unigrams. This result suggested that we should lemmatize unigram features and eliminate their inflexions. Therefore, in this thesis we use GPoSTTL tagger to lemmatize reviews. In our work, we do not use a stemmer to process inflexed verbs, adjectives and adverbs due to its intrinsic problems. These problems will be discussed in Chapter 4.

Furthermore, one of Mullen's conclusions is that SVM is an appropriate algorithm for the binary sentiment classification problem. As a whole, Mullen's method is a relatively complete solution to date, so in this thesis we also use SMO classifier provided by WEKA, and compare the effectiveness of incorporating above four features: Average_PMI, Average_lesk, Average_hso, Average_jcn with using sentiment terms alone.

3.2.3 Rating inference by exploiting class relationships

Bo Pang and Lillian Lee [75] addressed the N-star rating problem with respect to a multi-point scale (e.g., one to five stars). They recognized the important differences between standard multi-class text categorization and rating inference, that there are several different degrees of similarity between class labels.

To make use of the variant similarities between class labels, Pang and Lee applied a meta-algorithm, based on a *metric labeling* formulation of the problem that alters a given n-ary classifier's output in an explicit attempt, to ensure that similar items receive similar labels.

They implemented three algorithms, including *One-vs-all* (OVA) (Rifkin and Klautau, 2004), *linear, ε-insensitive* SVM regression (REG) (Vapnik, 1995; Smola and Scholkopf, 1998) and *metric labeling* method (PSP) (Kleinberg and Tardos, 2002), and used these algorithms to compose their supervised meta-learners.

Actually, Pang et al. did not implement full-star rating task directly. To avoid the complexity of imbalanced data problem, Pang and Lee simplified the five-star annotation by folding these minority classes into adjacent classes, thus arriving at a four-class problem; furthermore, they even reduced the five star rating into a three-class task in which the categories 0, 1, and 2 are essentially "negative", "middling", and "positive", respectively.

Table 3.8 shows the experimental result by Pang's methods:

	Significant differences, three-class data				**Significant differences, four-class data**			
	ova a b c d	ova+PSP a b c d	Reg a b c d	reg+PSP a b c d	ova a b c d	ova+PSP a b c d	Reg a b c d	reg+PSP a b c d
ova		↑ ↑ ↑ ▪	← ← ← ←	▪ ← ▪ ▪		▪ ↑ ↑ ↑	↑ ↑ ▪ ▪	↑ ▪ ▪ ↑
ova +PSP	◀ ◀ ◀ ▪		← ← ← ←	← ← ← ▪	▪ ◀ ◀ ◀		↑ ▪ ▪ ▪	↑ ▪ ▪ ▪
reg	↑ ↑ ↑ ↑	↑ ↑ ↑ ↑		▪ ↑ ▪ ↑	← ← ▪ ▪	← ▪ ▪ ▪		▪ ▪ ▪ ▪
reg +PSP	▪ ↑ ▪ ▪	↑ ↑ ↑ ▪	▪ ◀ ▪ ◀		← ▪ ▪ ←	← ▪ ▪ ▪	▪ ▪ ▪ ▪	

Triangles point towards significantly better algorithms for the results plotted above.

Specifically, if the difference between a row and a column algorithm for a given author dataset (a, b, c, or d) is significant, a triangle points to the better one; otherwise, a dot (.) is shown. Dark icons highlight the effect of adding PSP information via metric labeling.

Table 3.8 Results for main experimental comparisons.

Table 3.8 summarizes the average 10-fold crossvalidation accuracy results of Pang's experiments. The three algorithms described above definitively outperform the simple baseline of predicting the majority class, although the improvements are smaller in the four-class case.

Interestingly, the data was distributed in such a way that the absolute performance of the baseline itself does not change much between the three- and four-class case (which implies that the three-class datasets were relatively more balanced); furthermore, the Author's datasets seem noticeably easier than the others. They examined the effect of implicitly using label and item similarity. In the four-class case, regression performed better than OVA; but for the three-category task, OVA significantly outperformed regression for all four authors selected by Pang et al.

In the four-class case, metric labeling and regression seem roughly equivalent. Pang and Lee attributed this result to the fact that the relevant structure of the problem is already captured by linear regression (and perhaps a different kernel for regression would have improved its three-class performance). However, a ccording to additional experiments they ran in the four-class situation, the test-set-optimal parameter settings for metric labeling would have produced significant improvements, possibly indicating there may be greater potential for their framework.

In Pang and Lee's experimental setup and result, there are some very important aspects that need to be paid attention to:

1. Ordinal Information

   In Pang's method, they took a *regression* perspective by assuming that the labels come from a discretization of a continuous function g mapping from the feature space to a metric

space. The metric space was proposed for using the ordinal feature hidden in the labels; We think the utilization of the important ordinal information was not adequately utilized by simply adopting *linear, ε-insensitive* SVM regression (REG). Therefore, we issued another methods in our experiments based on output engineering.

2. The rationality for using regression method

   Undoubtedly, the appropriate role of a regression method in full-score (five star) prediction is still an open issue. Actually, it is doubtful and debatable that the five-star rating scheme could be described or expressed as a series of discrete numbers at identical interval. In other words, we are not convinced that a linear model should be used and how to use it properly in this setting.

3. The correlation between the movie reviews and their scores

   Like other sentiment anlysis research, this experiment paid little attention to the correlation between linguistic clues and semantic sentiments. Unfortunately, as metioned above in section 3.1.3, and shown in Table 3.3, movie reviews hold the lowest correlation, 0.3608, between the sentiment orientation of reviews and the star rating result. In other words, it is hard to evaluate the true accuracy of five star annotations because of the loose relationship between the review text and its sentiment score.

   Therefore, we use the Bank Review data which hold a highest correlation value, 0.6167 (as shown in Table 3.3), between their plain text and their scores.

4. The calibration of different authors' scales

   Pang's experiments were built on movie reviews by four different authors. Their datasets had 5006 labeled reviews and only one author each. Therefore their work was not relevant to settings with many authors but very little data for each; in other words, their four datasets are divided by different authors, and are not overlapped with each other.

The reason why Pang and Lee adopted documents written by the same author in each dataset is to factor out the effects of different choices of methods for calibrating authors' scales. Pang et al's decision is related the open issue in the sentiment analysis field that "Sentiment and what we can detect from text are not the same" (by Michael Gamon [76]). Because the languages and cultures differ in the expression of sentiment, calibrating different authors' scales and detecting their habits in their own expressions are challenging tasks. Natually, Pang et al. chose simplifying the complexity to avoid the difficulties.

However, Pang's simplification solution, to some extent, affected the generality of their research. For presenting the significance of a method, the generalization is a basic prerequisite. Therefore, generalizing rating-inference to mixed-author situations is an important work, because without an evaluation on the whole corpus, actually it is not comparable to the results reported by some other research.

There are some different possible methods to avoid the limitation of Pang et al's research and expand the supervised learning to a wide range of authors. For example, using an independent model to determine the author-independent characteristics, and using the author-independent characteristics to build related features into the feature set.

In our experiments, we created a collection of 3164 bank reviews by different authors, and assumed that the rating conversions mapped correctly into a universal rating scheme.

Chapter Four

Methodology

Quantitative sentiment analysis is challenging work because there is little previous work focusing on quantitative sentiment rating or multiclass classification with a five star rating scheme. Because of the lack of previous references in this area, this thesis practices extensive explorations in the following four aspects:

A. feature selection using GI, SO-PMI-IR and WordNet

B. making use of ordinal information via meta learning

C. dealing with imbalanced data by re-sampling and other approaches

D. combining multiple models

Our quantitative evaluation of subjective opinions is twofold, and we will implement both unsupervised learning and supervised learning algorithms in this thesis. Our objective is to determine the effect of different types of features (unigrams, feature set 1, feature set 2, and feature set 3) and observe how different classifiers can perform and be modified to obtain as high as possible accuracies for five star score prediction. In different phases of our machine learning task, we are trying to solve different specific problems we encounter, improve the performance, and achieve a better understanding to each type of features. Finally, we combine the best unsupervised model and the best supervised model together, and achive better performance than either of them alone. In detail, the essential works of our experiments include:

First, in the feature selection step, we are:

a) Using WordNet relatedness as one measure of feature selection instead of using

SO-PMI-IR method alone. Furthermore, we expand the feature set by utilizing all substantives - nouns, verbs, adjectives and verbs - instead of adjectives only.

In the experiments, we start from unigrams and gradually expand our feature sets by incorporating bigrams, SO-PMI-IR selected terms, and WordNet selected terms with initial unigrams, and investigate which feature set is more effective than others.

b) Adding topical synthetic features of Average_PMI, Average_lesk, Average_hso, Average_jcn into the original feature set which include only lemmas and bigrams, to investigate the effect of synthetic features.

Secondly, we explore the performance of our unsupervised learning algorithm and four supervised learning algorithms: BayesNet, Naïve Bayes, C4.5, and SVM.

Thirdly, we adopt a simple meta-learning solution to make use of ordering information of five star score labels. In addition, because the imbalance of the data seriously and adversely affects the performance of classification, we use relative filtering approaches to alleviate its negative impact.

Finally, we combine unsupervised learning and the best supervised classifiers to increase predictive performance over these single models.

4.1 Dataset Characterization and Preprocessing of Reviews

This section describes the general nature of the datasets used in this thesis as well as the methods used to prepare the data for experimentation.

4.1.1 Understanding the Data of Bank Reviews

The dataset is extracted from www.epinions.com. This website has also been used as the data source in previous research by P.D.Turney [14].

Our sentiment analysis is based on 3164 reviews of the 46 banks from Epinions (www.epinions.com). All collected reviews were written by Dec 30 2007 and are reviews of banks. Each of them was written by a different unprofessional author; any person with a Web browser can become a member of Epinions and contribute a review. For the effectiveness and integrity of the data, we only collect reviews from banks which have no less than 10 reviews. The Figure 4.1 illustrates the format of reviews before they were transformed from HTML into plain text:

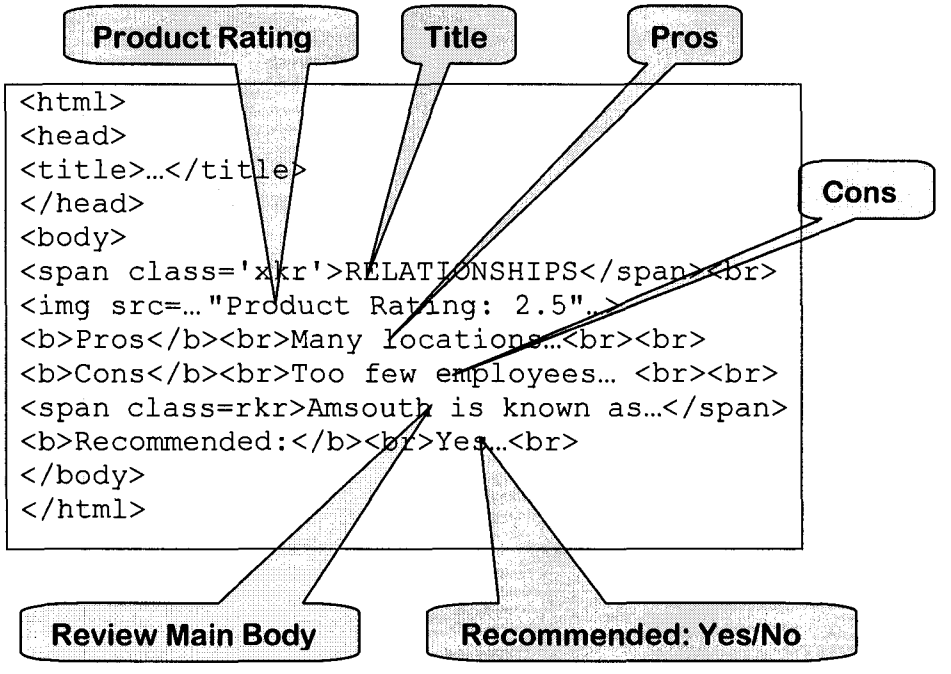

Fig 4.1 The Format of HTML Reviews

As shown in Figure 4.1, all reviews comprise a *Title*, a *Pros*, a *Cons*, a *Product Rating*, a *Review Main Body*, a *Recommended* field, the *writer's name* and the *date of the review*. We use

Perl to automatically remove the HTML tags and punctuation from the downloaded HTML review files, extract the textual contents of *Title*, *Pros*, *Cons* and *Review Main body* fields of each review as the raw material, and finally write the raw material into plain text review files which are used as the corpus for feature selection. Aside from the textual content, we also obtain the score value from the "*Product Rating*" field, which is used as the label of samples for training.

The *Recommended* field is automatically removed because it is instructive only in binary classification rather than in the five star rating system. Similarly, the fields of *writer's name* and the *date of the review* are deleted because they do not provide information about customers' sentiment orientation.

4.1.2 Lemmatization and Tagging (Preprocessing of Reviews)

As illuminated in section 4.1.1, we obtain the the plain text of reviews as the result of the first step of preprocessing. The plain text of reviews then acts as the input to the lemmatization. We use GPoSTTL to tag the lemmas with POS tags, and lemmatize words from their inflexion to their root format.

With the inflected forms of words, we cannot count terms accurately. On the other hand, the stemmer does not fulfill the requirement of recovering the stemmed words to their root form. For example, the stemmed result of the past tense 'produced', by the Porter Stemmer, is 'produc' instead of 'produce'. Unfortunately, with this stemmed form of words, take 'produc' as the example, we cannot extract the sentiment terms 'produce' from GI, and we will fail to extract the entry of 'produce' from GI. Therefore, we use lemmatization and not stemming in our experiments.

Therefore, we use GPoSTTL to lemmatize the reviews. GPoSTTL is an enhanced version of Brill's rule-based Parts-of-Speech Tagger, with built-in tokenizer and lemmatizer. It is

developed as an open-source alternative for a Penn Treebank tagger, so its tag set is compatible with the Link Grammar Parser which is used to capture bigrams in the section 4.2.2 and 4.2.3.

There are a total of 44 Penn POS tags appearing in the lemmatized result of the 3164 reviews. Because WordNet has only these four kind of substantives, we need to map these 44 Penn POS tags to 'noun' ,'verb' ,'adjective' ,'adverb' respectively when calculating the relatedness between sentiment words by WordNet::Similarity package. Therefore, we build a mapping between the Penn POS tags and the WordNet POS tags. This list is shown in Appendix B.

Lastly, we strip stop words from the lemmatized tagged reviews. For eliminating regular stop words, we combined two resources from [35] and [36]. Furthermore, we need to filter out some special high-frequency uninformative words such as 'Mutual' or 'Washington' which are component used for composing the bank names. These words help neither classification nor unsupervised learning, but they may result in excessive overfitting.

4.1.3 Auxiliary Preprocesing Phases and Approaches

There are some auxiliary preprocessing phases were adopted in previous works of sentiment analysis. Although these preprocessing steps are not necessary for this thesis, they indicate critical challenges in sentiment analysis tasks. We tested these approaches during the preprocessing step, but due to various reasons we do not adopt them in our final experiments, however they may benefit future works or improve the experimental results on different corpus or testbeds. Therefore, we list these approaches as follows, and explain the reason why they are not used in this thesis. They mainly focus on two typical issues:

a. (Subject, Sentiment) Association by Relationship Analysis

As mentioned in section 3.1.4, J.Yi, T.Nasukawa, R.Bunescu and W.Niblack [12]

proposed topic association method to solve the problem that different sentiment expressions might discuss different subjects in the same review.

However, according to our preliminary study, we found in our bank review testbed, the the opinions of banks and the opinions of services/products are commonly consistent. Therefore, we do not distinguish the opinions for different topics or subjects.

b. Distinguishing Objective from Subjective Statements

Many reserchers concentrate their works on distinguishing objective from subjective sentences when classifying reviews. The important recent works include:

1. Riloff and Wiebe [69] proposed a bootstrapping process that learns linguistically rich extraction patterns for subjective expressions. The learned patterns are then used to identify more subjective sentences.

2. Subsequently, Wiebe et al [70] presented methods for extracting subjective expressions from corpora relying on subjectivity clues such as low-frequency words, collocations, and adjectives and verbs, which are identified using distributional similarity.

3. Based on the assumption that objective and subjective sentences are more likely to appear in groups, Pang and Lee [71] presented their method of distinguishing objective statements from subjective statements. They built a manually-annotated subjectivity data set, and then use Naïve Bayes classifier to predict if a sentence is subjective or objective.

4. A similar experiment is presented in Yu and Hatzivassiloglou [72]. They extended the above approach by (1) taking into account non-adjectival parts-of-speech, (2) using larger sets of seed words and (3) including unigrams, bigrams, trigrams, part of speech information, and polarity into the feature set.

They also combine multiple Naïve Bayes classifiers for the same task, where each Naïve Bayes classifier focuses on a different part of the feature set.

We do not implement any methods Distinguishing Objective from Subjective Statements due to following reasons:

1. The improvements provided by distinguishing objective from subjective statements were not statistically significant, because matching the verb and its subject in a sentence is difficult, even sometimes the object is wrongly recognized as the subject of a sentence. Generally, this work involves complicated entity recognization which is a challenging area of text mining. When analyzing complex n-grams especially in case that the subject and verb are not adjoining, we need to implement a series of time-comsuming preliminary trial experimens to determine an appropriate *window length.* Due to the conflict of our limited time and the complexity of entity recognization, we leave this part of tests in future works.

2. The accuracy of sentiment classification closely relates to the number of sentiment phrases. However, distinguishing objective from subjective sentences might hamper extracting some correct sentiment terms due to the above reason 1. In fact, the sentiment phrases are very sparse already, so the errors in the distinguishing step usually has a slightly harmful effect, or even sometimes causes the learning performance to deteriorate rapidly.

4.2 Feature Selection

In this thesis, we adopt corpus-based automatic feature selection rather than manual or semi-manual feature selection. Section 4.2.1 illuminates the similarities and differences between these two feature selection strategies, compares their merits and defects, and discusses the reason why corpus-based automatic feature selection is chosen over manual methods.

We create four feature sets for our experiments in chapter 5, including unigrams, the Feature Set 1, the Feature Set 2, and the Feature Set 3. Throughout the rest of this section, from section 4.2.2 to 4.2.5, we present the motivation, heuristics, and methodolody of building these four feature sets respectively.

4.2.1 Manual and Corpus-Based Automatic Feature Selection

For sentiment analysis, there are two types of methods for feature selection:

a. Manual or semi-manual feature selection

   At the preliminary stage of SA, most research on sentiment-based classification has been at least partially knowledge-based. Some of this work focuses on classifying the semantic orientation of individual words or phrases, using linguistic heuristics or a pre-selected set of seed words (Hatzivassiloglou and McKeown [13]; Turney and Littman [21]).

   Past work on sentiment-based categorization of entire documents has often involved either the use of models inspired by cognitive linguistics (Hearst [38]; Sack [39]) or the manual or semi-manual construction of discriminant-word lexicons (Huettner and Subasic [54]; Das and Chen [40]; Tong [41]).

   As discussed in section 3.1.7, Bo Pang et al [10] had shown problems with this method. In their preliminary study, they found that humans' intuition cannot always effectively choose sentiment words.

b. Corpus-based automatic feature selection

   As mentioned in section 3.1.3, P.D.Turney selected in-domain features in his unsupervised learning using SO-PMI-IR methods, and discussed the weaknesses of previous knowledge-based manual feature selection [14]. His work is an early example of

corpus-based feature selection. Turney pointed out that in open-domain feature selection, each phrase must be manually added to a special lexicon and manually tagged as indicating positive or negative sentiment. The lexicon is specific to the domain and must be built anew for each new domain.

In addition, Bo Pang [14] used a simple method to select high-frequency unigrams (at least four times) and bigrams (at least seven times) as features. In their preliminary examination, they found that the corpus-based technique is way better than intuitions or manual methods.

Moreover, Andrew Lacey [20] also emphasized that sentiment data for varying domains can be quite different, and he supports the extraction of features strictly from the given domain.

From then on, based on the above preliminary experiments, most of sentiment analysis tasks used corpus-based feature selection. In this thesis, we explore the corpus-based techiniques rather than relying on intuitions or semi-manual approaches, to select good indicator features.

4.2.2 Unigrams Selected using General Inquirer

4.2.2.1 Motivation and Heuristics

In the paper for predicting semantic orientation by Hatzivassiloglou and McKeown [13], they designed their algorithm for isolated adjectives, rather than phrases containing adjectives or adverbs. This has been discussed in more detail in Section 3.1.1, along with other related work. We found that although adjectives alone are too limited to provide adequate sentiment information, they still play the most important role for expressing subjectivity. Although Hatzivassiloglou's feature set seems too limited, undoubtedly to some extent the good

performance of their exprtiments took advantage of unigrams.

On the other hand, nouns, verbs and adverbs also bring rich sentiment information to our sentiment analysis, compensate for the defects of Hatzivassiloglou's method which uses only adjectives as features, and benefit from sentiment terms belonging to other POS, we expand the feature scope to all the four types of substantives: adjectives, adverbs, nouns and verbs.

Moreover, Bo Pang et al concluded in their paper [10] that unigrams alone outperformed mixed unigrams and bigrams. This conclusion strongly suggests that unigrams may contain richer sentiment clues than n-grams. Therefore it is worth comparing the performance of a unigram feature set and a mixed feature set of both unigrams and bigrams.

Therefore, at first, we use General Inquerer dictionary (hereafter GI) to select the unigrams and form our basic baseline feature set.

4.2.2.2 Methods for Generating Unigrams

Based on lemmatized plain text files of reviews, we select high-frequency sentiment nouns, verbs, adjectives and verbs which are marked as 'Positiv' or 'Negativ' in General Inquirer dictionary (hereafter GI). We limited the unigrams to the lemmas appearing at least five times in the 3164 reviews, because the low-frequency unigrams cause the learning process to be very slow while they actually do not improve the results significantly.

As a result, we extract 1447 unigrams which must be seen at least five times in corpus. Because we do not consider punctuation as effective sentiment terms, we filter out all punctuation from our feature sets. Furthermore, we ignore stop words included in the stop word list [35] and [36], wrongly tagged or lemmatized terms, all numbers, and special characters such as currency characters.

There is a potential problem that might seriously affect the learning process. As described in section 4.2.1, we adopt corpus-based feature selection because this method takes full advantage of using domain-specific features; In other words, a good set of sentiment features for a given domain is defined functionally rather than superficially like those defined by experts or dictionaries. However, we need to avoid a kind of non-sentiment proper noun words and other similar words. For example, using the banks names such as "Washintong Mutual Bank", "Citybank", "Union bank" as features, either for unsupervised learning or supervised learning, indeed is a factor of excessive overfitting. Therefore, in feature selection, we pay much more attention to the words that begin with capital letter than to other words, and actually delete all words containing capital letters. Of course, we ensure that the first word in a sentence get rid of this operation, if it does not belong to the proper noun list.

4.2.3 Feature Set 1: Phrases Selected using GI and SO-PMI-IR

In this section, we build our Feature Set 1 based on sentiment phrases consisting of unigrams (lemmas) and bigrams.

4.2.3.1 Motivation and Heuristics

During the first phase of our preliminary study and experiments, we found that only relying on GI resulted in the omission of some important sentiment terms. For example, 'inept' is a high-frequency sentiment words used in bank reviews, to express a strong negative unsatisfactory feedback to bank services, but are absent in our unigrams feature set. Some other absent sentiment words include, for example, the obvious negative words like 'expletive', 'inexcusable', 'unrealistic' and positive words like 'beautifully', 'gentleman', 'satisfying'. Obviously, using only General Inquirer fails to find some useful sentiment words. One reason causing this problem is that some of those words are marked as neither '*Negativ*' nor

'*Positiv*' in GI; another reason is that some of them are not collected by GI. Therefore, we need to look for additional solutions.

Therefore, we have to look for alternative sentiment term extraction solutions, aside from GI, to avoid the loss of representative sentiment features. Natually, we first chose Turney's SO-PMI-IR algorithm. We setup an appropriate SO-PMI-IR threshold, and found new sentiment words whose SO-PMI-IR score are higher than the positive threshold or fewer than the negative threshold.

In addition, Feature Set 1 does not include unigrams alone. As described in section 3.1.7, according to Bo Pang [10], arbitrarily selected bigrams are not as useful as the unigram presence is. His conclusion implied that bigrams may cause the learning accuracy to decline. On the contrary, Alistair Kennedy and Diana Inkpen [11] determined that valence shifters significantly improve the binary classification of reviews. For comparing the effect of unigrams and mixed unigrams and bigrams, we involved bigrams in our Feature Set 1. Our bigrams extraction pattern is derived from Turney's paper [14]. In Turney's method, he used the patterns shown in Table 1.1. To avoid the extremely complex linguistic analysis for n-gram collocations, we reconstruct Turney's bigram pattern. Similar to the methods used by by Alistair Kennedy and Diana Inkpen [11], we extracted only consecutive words as bigrams and limited bigrams to the range of "modifiers+adjectives", in which one member of the pair is an adjective and the precursor is a modifier, as shown in Table 4.1.

	First Word	Second Word	The tool used to capture patterns
1.	RB,RBR, or RBS	JJ	GPoSTTL
2.	JJ	JJ	GPoSTTL
3.	NN or NNS	JJ	GPoSTTL
4.	VP	ADJP, ADVP	Link Grammer Parser

Table 4.1 Reconstructed patterns for extracting two-word phrases

Aside from unigrams selected by GI, all additional unigrams and bigrams are selected by

SO-PMI-IR algorithm. We programed the SO-PMI-IR method based on Google API, relying on the 'AND' operator provided by Google Development Team. We discuss our SO-PMI-IR algorithm in detail in next section.

Finally, we incorporated these new sentiment unigrams and bigrams with GI-selected unigrams, and created the Feature Set 1.

4.2.3.2 Methods of Generating Feature Set 1

Based on the Google API, we build our own SO-PMI method which is used to add more positive and negative phrases into our feature sets. This SO-PMI algorithm is derived from P.D.Turney's SO-PMI-IR method [14]. Relying on the Alta Vista search engine, Turney's SO-PMI-IR algorithm uses 'NEAR' to calculate the SO score of a sentiment word. The'NEAR'operator is used to count the number of hits about one phrase near the seed words. However, Alta Vista no longer supports 'NEAR', so we use the'AND' function of the Google API instead. In addition, differing from P.D.Turney's SO-PMI-IR method, our SO-PMI algorithm take into accout both unigrams and bigrams.

We use the following formula to calculate the SO score of a phrase.

$$SO(phrase) = \log_2 \left[\frac{hits(phrase\ AND\ p\_query)\ hits\ (n\_query)}{hits(phrase\ AND\ n\_query)\ hits\ (p\_query)} \right] \qquad \textbf{(4.1)}$$

In formula (4.1), the positive and negative reference terms '*p_query*' and '*n_query*'are defined as:

$$p\_query = p\_seed\_word_1\ OR\ \ldots\ OR\ p\_seed\_word_{10} \qquad \textbf{(4.2)}$$

$$n\_query = n\_seed\_word_1\ OR\ \ldots OR\ n\_seed\_word_{10} \qquad \textbf{(4.3)}$$

Both '*p_query*' and '*n_query*' are respectively the set of top ten most frequent sentiment seed words, which are selected by GI, and are considered to be the most representative sentiment words. The ten positive and ten negative seed words are shown in table 4.2:

	p_seed_word	n_seed_word
1	good	problem
2	free	bad
3	great	negative
4	friendly	hard
5	easy	terrible
6	happy	poor
7	helpful	ridiculous
8	nice	difficult
9	fine	horrible
10	right	outrageous

Table 4.2 The seed words used for SO-PMI-IR

When substituting (4.2) and (4.3) for '*p_query*' and '*n_query*' in (4.1), we execute the following transformations:

$$\begin{aligned} hits(phrase\ AND\ p\_query) &= hits(phrase\ AND\ (p\_seed\_word_1\ OR\ \ldots\ OR\ p\_seed\_word_{10})) \\ &= hits((phrase\ AND\ p\_seed\_word_1) + \ldots + (phrase\ AND\ p\_seed\_word_{10})) \end{aligned}$$

$$\begin{aligned} hits(phrase\ AND\ n\_query) &= hits(phrase\ AND\ (n\_seed\_word_1\ OR\ \ldots\ OR\ n\_seed\_word_{10})) \\ &= hits((phrase\ AND\ n\_seed\_word_1) + \ldots + (phrase\ AND\ n\_seed\_word_{10})) \end{aligned}$$

Hence, the *hits* (*phrase AND p_query*) is substituted by the sum of the co-occurrence of the phrase and single positive seed word together, and *hits* (*phrase AND n_query*) likewise.

In this formula, the number of hits of a phrase AND the seed word are obtained by SOAP::Lite package in the Perl language which is used to count the cached pages and number of hits meeting the AND condition.

For determining the Semantic Orientation of words, we adopt the SO-PMI 1 method from the paper of Alistair Kennedy and Diana Inkpen [11]. First, we calculate the SO-PMI scores of all positive and negative words in GI, compare their prediction result of SO-PMI and the '*Positiv*' and '*Negativ*' value in GI, and obtain the threshold which maximizes the accuracy of sentiment orientation prediction. In our experiments, when we set the positive and negative threshold equal to 0.768 and -0.353 respectively, the SO prediction result of these sentiment terms achieved the highest agreement 92% and 91% with the '*Positiv*' and '*Negativ*' value in GI. Therefore, finally our threshold for positive words is 0.768, and -0.353 for negative terms.

In this way, given the ten pairs of sentiment seed words in Table 4.2, we can predict whether the given words are positive or negative in sentiment. We implement our SO-PMI method on the terms which appear more than 3 times and do not have an entry in GI, and add them into our feature set 1 if and only if their SO-PMI score is no lower than 0.768 or no higher than -0.353. Finally, we have total 1447 features in Feature Set 1 for unsupervised learning, in which 1057 are selected by GI, the other 390 are selected by the SO-PMI algorithm.

We ignore the noun phrases (line 1, Table 1.1) and verb phrases (line 5, Table 1.1) from Turney's patterns, because they do not provide rich sentiment information. However, Turney's patterns are weak at detecting some "modifiers + adjectives" bigrams, especially the "negation + adjectives" pattern. In the paper by Alistair Kennedy and Diana Inkpen [11], they emphasized that negation terms contribute a lot in sentiment classification, so we added a new negation pattern (Line 4, Table 4.1) which is extracted using the Link Grammar Parser.

Neglecting negation modifiers often results in a misunderstanding of customers' sentiment orientation. For example, one may comment that "*Our checking account opening experience at First Union was **not pleasant**.*" If the 'not' is not detected, this sentence will be judged as positive, and the favorable orientation of 'pleasant' will present a positive opinion. To correct this kind of errors, we use the Link Grammar Parser to capture the negation of the expression.

```
++++Time                                          0.02 seconds (38.43 total)

Found 17 linkages (17 with no P.P. violations)

  Linkage 1, cost vector = (UNUSED=0 DIS=0 AND=0 LEN=22)


    +-----------------------------------------Xp-----------------------------------------+
    +-----------Wd-----------+---------------------Ss--------------------+               |
    |      +--------Ds-------+         +--------MVp--------+----Js---+     +-----Paf----+     |
    |      |       +----A----+----Mg---+-----Os----+       |   +--G--+     +-EBm+       |     |
    |      |       |         |         |           |       |   |     |     |    |       |     |
LEFT-WALL our checking.g account.n opening.v experience.n at First Union was.v not pleasant.a .

Constituent tree:

(S (NP (NP Our checking account)
       (VP opening
           (NP experience)
           (PP at
               (NP First Union))))
   (VP was not
       (ADJP pleasant))
   .)
```

Fig 4.2 Negation Reorganization by Link Grammar Parser

As Shown in Figure 4.2, once the 'EB' (a "be" before an object, adjective, or prepositional phrase),'N' (a "not" preceding auxiliaries),'NT' (connection between 'not' and 'to') links in the linkage graph, we capture the adjoining 'VP' and 'ADJP' or 'VP' and 'ADVP' structures from the constituent tree following the linkage graph.

Furthermore, because searching and parsing n-gram phrases is time-consuming, and the effective scope of modifiers is hard to determine accurately, in our experiments, we extract only adjoining pairs of words (bigrams) from reviews.

Based on the parsed documents by GPoSTTL or Link Grammar Parser, we can find the bigrams that match the patterns shown in Table 4.1. As presented in Table 4.1, the 'JJ', 'NN', 'RB', and 'VB' respectively denote adjective, noun, adverb, and verb.

Take the line 1 of Table 4.1 as an example. That pattern denotes a bigram consisting of an adverb followed by an adjective. If this adverb is in the GI dictionary, and is marked as '*Ovrst*' (overstatement), or marked as '*Undrst*' (understatement), it is an effective sentiment bigram.

In Table 4.1, the fourth pattern is different. This pattern is used to extract negation phrases. When we find '*no*' or '*not*' which is used to construct VP phrases, we check whether there is an adjective in the following ADJP or ADVP. This way, we obtain the negation phrase.

4.2.4 Feature Set 2: Adding unigrams using SO_WN algorithm

In this section, we build our Feature Set 2 by adding unigrams using WordNet relatedness score SO_WN.

4.2.4.1 Motivation and Heuristics

In section 4.2.3, using SO-PMI-IR algorithm, we incorporated the bigrams and some important missing sentiment unigrams into the Feature Set 1. Unfortunately, there are still three obvious weaknesses of the SO-PMI-IR algorithms:

1. The SO-PMI-IR algorithm estimates PMI relying on the fake hits number returned by issuing a query through Google API, so it is neither accurate nor stable enough. This wrongly counted hits number is caused by repeated mirror links cached by the Google search engine.

   The SO-PMI algorithm estimates PMI by issuing a query through the Google API to obtain the number of hits which is the number of matching documents cached by Google. However, one identical webpage may have many cached mirror links in Google and any other ISPs. For example, in the first returned page, we can see 204,000 links belonging to

20,400 pages when issuing "Washington mutual customer review", but we find only 110,480 real unique links in 11,048 pages when going through to browse all returned pages. This is because many Internet Service Providers (hereafter ISPs), including Google, do not filter out repeated links when returning the searched hits, especially for single word query, such as *hits*("*good*"), the returned hits number is often many times greater than the real hits number.

2. Relying on the frequency with which the Google index is updated, the query result of hits number varies at frequent intervals; in other words, the query result is inconstant actually. This situation undoubtedly further decreases the accuracy of SO-PMI-IR algorithm.

   The hits number returned by any search engine is only a rough estimation, not the exact count. If we perform two searches on Google, by sending a query at different times (even at intervals of few minutes), we often get two different numbers of hits. Generally, the hits number is generated from the Google index, but the results in the Google index changes regularly. These changes rely on updates to its index, including the addition of new sites and the removal of outdated links, plus a variety of ongoing, automated processes aimed at improving the quality and content of its search results.

   Consequently, we may not always see exactly the same results when querying Google by using either Web APIs or keying in a query into the search box on the Google website. Sometimes, this discrepancy may significantly affect the SO-PMI computation; for example, we get 189,000 hits when using the query "very rude AND quite impolite", but 163,100 hits 15 minute later. This unstable counting of hits results in a comparatively unreliable feature set using only SO-PMI algorithm.

3. Turney's SO-PMI method is based on the Pointwise Mutual Information between two words or two phrases. In PMI, the *p*(*word 1&word 2*) denotes the probability that word 1 and word 2 co-occur, and we use *hits*(*word 1 AND word 2*) to denote this measure in SO-PMI computation. However, using the number of hits of the co-occurrence of two words is only an approximation of *p*(*word 1&word 2*) either using *hits*(*word 1 AND word*

2) or *hits*(*word 1 NEAR word 2*); in other words, they are rarely absolutely equal, so both approximations are hardly accurate to represent the real PMI value. This fact results in the unavoidable intrinsic difference between the computed PMI and the pure theoretical PMI value.

Therefore, we incorporate new features with Feature Set 1 to construct our Feature Set 2 using the SO_WN algorithm based on WordNet relatedness score.

4.2.4.2 Methods of Generating Feature Set 2

By replacing half of the phrases in Feature Set 1 with new phrases selected by the SO_WN measure we developed based on the the WN-Similarity package, we obtain Feature Set 2. We used the WordNet Similarity Package 1.04 and WordNet version 2.1 to implement SO_WN algorithm. In the rest part of this section, we introduce the detail procedure of implementing our SO_WN algorithm.

In the WordNet Similarity (hereafter referred to as the WN-Similarity) package, there are a total of six measures of similarity and three measures of relatedness, all of which are based on the lexical database WordNet. We experimented using all nine of these measures, to test their performance for predicting the sentiment orientation of the 1447 sentiment words which are in both the unigram feature set and the GI dictionary. In all 1447 words, a total of 675 noun phrases and verb phrases are predicted by the similarity measures including *res, lin, jcn, lch, wup* and *path*, because, as expressed in section 2.7, only nouns and verbs can be judged by a similarity algorithm; On the other hand, because the measures of relatedness are more general in that they can be made across part of speech boundaries, and they are not limited to 'is-a' relations, they can be used to compute relatedness between any part of speech. Therefore, we can use three relatedness measure *hso, lesk* and *vectors* to judge all 1447 words.

First of all, we must determine which measures we compute SO_WN score with. Hence, we propose a test to compare all WN-Similarity measures: for each sense ws_i of the word w in the

1447 words, we calculate its similarity to each of the 20 positive seed words in the list of sentiment seed words shown in Table 4.3. We add up all the 20 similarity values of positive and negative seed words separately, and then subtract the negative sum from the positive sum. If the result is positive, then this word sense is positive, otherwise it is negative. Finally, we iterate the same procedure nine times by using each of nine WN-Similarity measure of semantic similarity. We then rank the results by the nine different WN-Similarity measures in Table 4.4, in which the measure achieving the highest prediction accuracy on all 1447 sentiment terms should be the most promising measure for feature selection. (For judging the correctness of the SO prediction, we compare the predicted results of words with their '*Positiv*' or '*Negativ*' attributes in GI; in addition, for the unigrams not from GI and all bigram phrases, we classify them manually in advance.)

All the sentiment seed words are listed in Table 4.3:

	Nouns		**Verbs**		**Adjectives**		**Adverbs**	
	words	**freq.**	**words**	**freq.**	**words**	**freq.**	**words**	**freq.**
positive	protection	216	offer	1013	worth	241	worth	6
	friend	262	clear	411	true	160	necessarily	21
	advantage	134	experience	156	nice	452	*forward*	59
	offer	286	hope	174	good	1179	free	3
	trust	123	resolve	174	free	1022	smartly	3
	experience	722	like	208	fine	243	better	51
	care	209	understand	298	better	585	greatly	5
	deal	256	care	310	great	756	readily	16
	good	128	consider	310	real	251	*real*	3
	free	375	save	409	able	503	luckily	49
	help	161	deal	551	friendly	497	friendly	48
	patience	320	verify	167	important	149	primarily	26
	great	316	help	639	excellent	219	promptly	77
	benefit	143	give	1637	main	155	satisfactorily	3
	home	577	accept	182	easy	597	honorably	3
	interest	831	inform	293	open	301	undoubtedly	4
	bonus	163	open	1515	willing	152	easy	4
	open	145	provide	441	helpful	404	correctly	215
	right	458	allow	434	happy	422	remarkably	7
	security	296	love	249	major	177	best	7
negative	complaint	236	hate	121	cheap	81	unexpectedly	5
	nightmare	116	owe	122	*frustrate*	61	*cheap*	1
	inefficacy	329	drop	126	expensive	72	dishonestly	1
	chase	184	hit	170	stupid	88	*smack*	1
	bad	131	need	1418	bad	780	costly	1

	problem	1490	steal	170	low	394	badly	26
	error	354	cut	93	terrible	141	gruffly	13
	terrible	80	bother	132	insufficient	62	low	12
	fraud	96	refuse	233	sorry	128	warily	1
	cost	316	cost	233	outrageous	78	trouble	1
	expense	68	complain	196	competitive	84	frantically	2
	trouble	190	fail	92	ridiculous	126	competitive	1
	anxiety	216	avoid	366	rude	203	*backward*	2
	concern	106	deny	101	incompetent	89	terribly	1
	lack	158	concern	118	poor	298	silly	7
	poorness	156	waste	84	negative	178	*ill*	3
	mistake	423	miss	161	agony	62	regardless	53
	hassle	183	cancel	300	difficult	137	hard	4
	horrible	88	lose	521	horrible	148	temporarily	17
	fault	191	worry	129	hard	270	horribly	1

Table 4.3 The 20 positive and negative seed words for each part of speech used to calculate SO_WN

From Table 4.3, we can find that there are a few mistakes marked in italic font. For example, the adjective'*terrible*' appears in the category of negative nouns. These errors result from the wrong POS tagging by lemmatizer GPoSTTL. There are a total of nine misclassified sentiment words in all 160 words, but all of them have the correct sentiment orientation categorization and so we can simply ignore these mistakes. We believe this set of seed words is reliable.

One thing that needs to be emphasized is that, for bigrams, we adopt the same method to calculate their SO as we do on single unigram. We treat them as one term, and use the AND operator to get the hits of their co-occurance with the seed words. In other words, we do not analyze the two words of bigrams separately with seed words, because such an approach involves collocation recognization and other relatively sophisticated works relating to complex processes of natural language processing. Therefore, we leave such analysis for our future work.

In this test, the *jcn* and *lesk* algorithms rank the two best in all nine algorithms. The accuracies of all nine algorithms are shown in Table 4.4:

		Algorithm	Total number of words	Correct predicted words	Accuracy
Similarity Measures	1	Res	675	331	49%
	2	Lin	675	290	43%
	3	Jcn	675	452	**67%**
	4	Lch	675	357	53%
	5	Wup	675	398	**59%**
	6	Path	675	182	27%
Relatedness Measures	7	Hso	1447	868	**60%**
	8	Lesk	1447	897	**62%**
	9	Vector	1447	463	32%

Table 4.4 The test result of WN-Similarity on 1447 words

Since the three measures *jcn*, *lesk* and *hso* gave comparable accuracy of 67%, 62% and 60%, we restricted our remaining experiments to them because they are more efficient than the other six measures of WN-Similarity. (In additionto these three measures, by *wup* we also obtain good accuracy, and we will use it in section 4.3.3)

This result is consistent with the paper of Diana McCarthy et al, 2004 [56] in which it was found that both *jcn* and *lesk* are efficient and provide the best results in previous experiments involving word sense ranking. As a result, from the existing nine similarity measures, we chose to use the *jcn* by Jiang and Conrath [57] measure and the *lesk* by Banerjee and Pedersen [58]. In addition, the *hso* performance is the third best, and we will use it to construct the *Average_hso* feature for Feature set 3.

According to our observation, the sentiment words generated by *jcn* and *lesk* were very different, and did not overlap much. On the other hand, combining phrases selected by *jcn* and *lesk* together, we can cover 85% (of the *recall* value) sentiment words in the test of Table 4.4. Therefore, by assignning different weights *a1* and *a2* to *SO_WN_jcn* and *SO_WN_lesk*, as described in Figure 4.3, we can combine them together to obtain the SO_WN score.

We devise the method SO_WN targeting to automatically compute the sentiment orientation score of word senses for subjectivity. The pseudo code of SO_WN Algorithm is shown in

Figure 4.3:

SO_WN Algorithm

Input: total 160 seed words in which each POS of noun, verb, adjective, adverbs includes 20 positive and 20 negative seed words

Input: phrase p_i

Output: SO_WN score

If (Input phrase p_i is a bigram)

Then

w_i= second word of phrase p_i

If the first word is identifier Then valence =3

If the first word is diminisher Then valence =1

If the first word is negation Then valence =-1

Else

w_i= p_i

valence =2

EndIf

Foreach word sense w_i do

SO_WN(w_i)=0

SO_WN_jcn =0

SO_WN_lesk =0

For j =1 to n do

PS_j = top n frequent positive seed words that has the same word sense with w_i

NS_j = top n frequent negative seed words that has the same word sense with w_i

End For

Foreach w_p in PS_j do

SO_WN_jcn+= jcn_sim(w_i,wp)

SO_WN_lesk+= lesk_sim(w_i,wp)

End Foreach

Foreach w_n in NS_j do

SO_WN_jcn-= jcn_sim(w_i,wn)

SO_WN_lesk-= lesk_sim(w_i,wn)

End Foreach

If w_i is noun or verb

Then

SO_WN(p_i)=a1* valence * SO_WN_jcn+a2 * valence * SO_WN_lesk

Else

SO_WN(p_i)= valence * SO_WN_lesk

EndIf

End Foreach

Fig 4.3 SO-WN Algorithm (n=20)

In the SO_WN algorithm, we set the number (the value of 'n') of seed words to 20.

The main idea behind our SO_WN Algorithm is that we can derive information about a word sense based on information drawn from words that have a similar distribution to that of the given word sense. This idea relates to the unsupervised word sense ranking algorithm described in McCarthy et al. [56], which used the information about words with similar distribution to predict corpus frequencies for word senses.

Our experiments and SO_WN algorithm are based on a common observation: In word sense disambiguation (hereafter WSD), the heuristic of choosing the most common sense is extremely powerful because the distribution of the senses of a word is often skewed. Many reserchers, e.g. McCarthy et al. [56] concluded that: "The first sense heuristic which is often used as a baseline for supervised WSD systems outperforms many of these systems which take surrounding context into account." Therefore, in our SO_WN algorithm, we use the first sense of words by default.

What is different from (McCarthy et al., 2004) [56] is our goal of estimating the subjectivity of a given word sense. Because all reviews have been lemmatized and tagged with the Penn tag set by the GPoSTTL tagger, the POS of each word sense of a unigram can be determined using the mapping list in Appendix B. Again, we use the 20 most frequent positive and negative seed words for each POS of noun, verb, adjective and adverb, as shown in table 4.3. Starting with a given word w_i of all 30048 lemmas of the 3164 reviews, we compute its similarity score SO_WN with seed words using the algorithms *jcn* and *lesk* of the WN-Similarity package.

The *lesk* measure can be used when computing relatedness between adjectives, and adverbs as well as between nouns and verbs, but *jcn* can only be used for nouns and verbs. Their difference is that *lesk* is applicable to lexical resources which do not have the hierarchical structure that WordNet does; however, *jcn* can only work in the hierarchical structure of 'is-a' relationship.

Therefore, we directly assign the *lesk* score SO_WN_lesk to SO_WN(w_i) for all adjectives and adverbs, because the *jcn* score is not available for them. On the other hand, we combined *jcn* and *lesk* score for nouns and verbs. We use *a1* and *a2* to weight the *SO_WN_jcn* and *SO_WN_lesk*. We experimented using a simple set of 200 nouns and 200 verbs to determine the value of *a1* and *a2*, and we found that *a1*=0.57 and *a2*=0.43 produces optimal results for predicting the sentiment orientation of these words.

In feature set 1, there are a total of 1447 features of which 792 are unigrams and the other 655 are bigrams. In Figure 5.4, the pseudo code of the SO_WN algorithm not only deals with unigrams, but also includes a special formula to calculate the SO_WN score for bigrams which is different from the formula for unigrams.

As shown in Table 1.1, the bigrams we extracted focus on adjectival phrases (line 1 to line 3) and sentiment words modified by negations (line 4). Therefore, we let w_i be equal to the second word of the bigrams. Afterwards, we use the method of valence shifters, which are used in the paper by Alistair Kennedy and Diana Inkpen [11] and derived from Polanyi and Zaenen [67].

We check whether the 'First Word', the first column in Table 1.1, is an intensifier, a diminisher, or a negation. If the 'First Word' is a word entry in GI, and is marked as *overstatements* or *understatements,* then we setup the variable 'valence' and assign a value to it as shown in Table 4.5:

Type of modifier	Valence
None	2
Diminisher	1
Negation	-2
Intensifier	3

Table 4.5 Valence Shifters

Overstatements are intensifiers, which increase the intensity of a positive/negative phrase, while understatements are diminishers, which decrease the intensity of that phrase. To allow for intensifiers and diminishers all positive sentiment terms in our system are given a value of 2. In other words, the initial adjectives and other sentiment phrases have the valence of 2 by default. If they are preceded by an intensifier in the same clause then they are given a value of 3. If they are preceded by a diminisher in the same clause then they are given a value of 1. Negative sentiment terms are given a value of -2 by default. These values were proposed by Polanyi and Zaenen (2004) in their linguistic analysis study.

As metioned above, in Feature Set 1, there are a total of 1447 features of which 792 are unigrams and the other 655 are bigrams. For comparing the performance of the SO-PMI selected features and the SO_WN selected features, we use the SO_WN selected features to replace half of the amount of unigrams and bigrams of Feature Set 1.

We implement the SO_WN algorithm on the top 2000 most frequent lemmas (unigrams) and bigrams not from Feature Set 1. From these 2000 unigrams and bigrams, we use the 396 top ranked unigrams, sorted in decreasing order of their SO_WN score, to replace the 396 unigrams of Feature Set 1 whose SO-PMI scores are ranked at the bottom. Afterward, from the 2000 phrases again, we choose the 327 bigrams whose SO_WN score ranked at the top to

replace the 327 bigrams of Feature Set 1 whose SO-PMI scores are ranked at the bottom.

Consequently, we obtain Feature Set 2 which has the same number of features as Feature Set 1. In Feature Set 2, there are 396 new unigrams and 327 new bigrams which are extracted by the SO_WN algorithm.

4.2.4.3 Two special problems when using the SO_WN algorithm

There are two particular situations that we need to pay attention to, because they seriously affect the accuracy of SO_WN algorithm.

First, the WN-Similarity package returns an excessively high score when calculating the similarity between seed words and themselves. For example, when computing the positive summary score for the noun 'stupidity' with the *jcn* algorithm, the returned result (using 10 seed words) is shown in Figure 4.4:

```
stupidity friend 0.0604185787767323
stupidity offer  0.0506588189723659
stupidity goodness   0.0593494871396401
stupidity interest   0.0649852767533113
stupidity experience 0.0725148987295942
stupidity right  0.089464181398241
stupidity free   0
stupidity security   0.0606709732681374
stupidity home   0.0582727943356647
stupidity happiness  0.0485519534632259
========================================
stupidity cost   0.0680959779892179
stupidity trouble    0.0782124622300849
stupidity stupidity 27240199.487762
stupidity complaint  0.0502428072976397
stupidity need   0.0592824682735424
stupidity problem    0.0610282704278744
stupidity error  0.0623495908919229
stupidity hassle 0.0566236729030602
stupidity mistake    0.0623495908919229
stupidity fault  0.061126589589805
========================================
score of stupidity is -27240199.4821864=0.564886962836913-27240200.0470734
```

Fig 4.4 exceptional SO_WN score for a seed word (by 10 seed words)

Actually, the sentiment orientation of the seed words, either positive or negative, was already known in advance from GI, there is no need to calculate their SO_WN scores. However, the SO_WN algorithm is also used to determine the sentiment score for phrases in the next section and chapter 5 for unsupervised scoring, so we have to declare this problem here beforehand.

As shown above in Fig 4.4, the SO_WN score of *stupidity* will be an unreasonable absolute negative value due to the exceptional high value of its *jcn* score. To resolve this problem, when computing the SO_WN score for all the 160 seed words, we only use the other 19 pairs of seed words to calculate their SO_WN by temporarily excluding them from the seed words (In the 20 antonyms of this seed word, we delete the one with the lowest frequency). In other words, we filter them out to avoid calculating the similarity value *jcn_sim(wp,wp)*, *lesk_sim(wp,wp)*, *jcn_sim(wn,wn)*, and *lesk_sim(wn,wn)*.

Second, it is noteworthy that a pair of words closely related with each other does not always have same sentiment orientation. For example, we found the following result returned by the *lesk* algorithm of WN-Similarity between the words 'hate', 'stupidity' and the 10 seed words:

hate	offer	7	stupidity	friend	22
hate	consider	3	stupidity	offer	10
hate	save	4	stupidity	goodness	12
hate	open	7	stupidity	interest	16
hate	deal	3	stupidity	experience	23
hate	provide	2	stupidity	right	45
hate	help	5	**stupidity**	**brightness**	**55**
hate	**love**	**15**	stupidity	security	29
hate	allow	6	stupidity	home	7
hate	give	9	stupidity	happiness	19
hate	cost	6	stupidity	cost	30
hate	complain	8	stupidity	trouble	22
hate	avoid	7	**stupidity**	**stupidity**	**566**
hate	need	9	stupidity	complaint	21
hate	hit	8	stupidity	need	15
hate	steal	5	stupidity	problem	32
hate	miss	7	stupidity	error	31
hate	cancel	2	stupidity	hassle	6
hate	**lose**	**11**	stupidity	mistake	31
hate	refuse	10	stupidity	fault	18

Fig 4.5 The problem of lesk algorithm between a pair of antonyms (by 10 seed words)

As shown in Figure 4.5, the relatedness score between antonyms are very high; the returned value is even larger than the maximum value between it and the seed words of same orientation.

Therefore, for getting a feasible SO_WN estimation, we should not add lesk values between w_i and any seed word to the summary result of SO_WN(w_i) if they are a pair of antonym. For each seed word w_i, our solution is that first we find the antonyms of w_i using the $synset->antonyms( ) function of Lingua::Wordnet package, and then we do not add their similarity with if they are in the list of seed words. (In the 20 synonyms of this seed word, we also delete the one with the lowest frequency)

4.2.5 Feature Set 3: Adding synthetic features

In this section, we build our Feature Set 3 by adding four synthetic features Average_PMI, Average_lesk, Average_hso, and Average_jcn.

4.2.5.1 Motivation and Heuristics

As described in section 3.1.8, Mullen et al [18] derived some additional feature types using using the method of Kamps and Marx [64] which uses WordNet relationships to derive three dimensions pertinent to the emotive meaning of adjectives. The three dimensions are *potency* (strong or weak), *activity* (active or passive) and the *evaluative* (good or bad) introduced in Charles Osgood's Theory of Semantic Differentiation [65].

Because these three Osgood values depend on the synonymy synset of WordNet and only adjectives and adverbs are organized in synonym synsets, Mullen et al. build the three values EVA (*evaluative*), POT (*potency*), and ACT (*activity*) on a list of 5410 adjectives.

According to Mullen et al [18], using the Osgood values discussed above, sentiment classification achieved the highest accuracy of classifying movie reviews by SVM. Obviously, this result suggests that topical and synthetic information indeed benefits the learning process, so relying on WordNet we introduced our own synthetic features Average_PMI, Average_lesk, Average_hso, Average_jcn in our experiments, and explore their effects to the learning results.

4.2.5.2 Methods for Generating Feature Set 3

In Mullen et al's paper [18], they derived other feature types using the method of Kamps and Marx [64] which uses WordNet relationship to derive three values pertinent to the emotive meaning of adjectives. The three values are *potency* (strong or weak), *activity* (active or passive) and the *evaluative* (good or bad) introduced in Charles Osgood's Theory of Semantic Differentiation [65]. However, these three measures are not appropriate to our situation, and to some extent their function is repetitive with that of the valence shifters. Therefore we developed our own synthetic features.

Average_PMI is the average value of all the 1447 sentiment phrases of Feature Set 2. According to the research by Peter Turney [14], and by Alistair Kennedy and Diana Inkpen [11], pointwise mutual information (PMI) is an effective and reliable indicator of the sentiment orientation of customer reviews, either in supervised learning or unsupervised learning algorithms. Hence, we divide the sum of the SO-PMI value $SO(w)$ of all the 1447 sentiment phrases by the total number of features, to calculate the value of Average_PMI.

Moreover as shown in Table 4.4 and discussed in section 4.2.4.2, *jcn*, *hso*, and *lesk* are three measures that performed very well for sentiment classification. Therefore we involve their average value as our synthetic features. Because the *jcn* value is a similarity measure, so it is based on the 'is-a' relation; and the in 'is-a' relation of WordNet do not cross part of speech boundaries, so *jcn* measure is limited to making judgments between noun pairs (e.g., cat and

dog) and verb pairs (e.g., run and walk). Therefore, our Average_jcn value is only for noun and verb of Feature Set 2. We divide the sum of the *jcn* value of all nominal and verbal phrases by the total number of nominal and verbal phrases, to obtain the value of Average_jcn.

On ther other hand, because *hso* (Hirst and St-Onge, 1998) and *lesk* (Banerjee and Pedersen, 2003) are not limited to 'is-a' relations, we applied both of them on all 1447 features. Similar to the algorithm of Average_jcn, we divide the sum of the *hso* and *lesk* value of all 1447 phrases by the total number of all features, to obtain the values of Average_hso and Average_lesk. In other words, these three values Average_jcn, Average_hso and Average_lesk are all average values in a document.

In our twofold experiments, Feature Set 1 and Feature Set 2 are used in unsupervised learning, while all Feature Set 1 through 3 and the unigrams feature set are used in supervised learning.

4.3 Unsupervised learning

The objective of this thesis is to quantitatively evaluate the degree of customers' satisfaction from their on-line reviews, and to observe the effect of supervised learning and unsupervised learning methods respectively. In this section, we explore the method of unsupervised learning, and in next section we will concentrate on supervised learning.

Due to the highly domain-specific nature of the sentiment classification task, moving from one domain to another typically requires the acquisition of a new set of training data. For this reason, unsupervised or very weakly supervised methods for sentiment classification may be especially desirable. Our focus in this section, consequently, is on methods that require very little data annotation.

Many previous works using unsupervised learning have achieved very good performance. Hatzivassiloglou and K. R. McKeown 1997 [13] used unsupervised learning methods in

semantic orientation prediction based on 21 million word from 1987 Wall Street Journal corpus, and reported 92% accuracy on the classification task on 1336 adjectives; Michael Gamon and Anthony Aue [17] reported 73.95% accuracy using unsupervised method to identify sentiment vocabulary. The results of the above two experiments are consistent with P.D.Turney's [14] conclusion, in which Turney affirmed that using a complex algorithm does not readily ourperform using simple PMI-IR method in sentiment classification.

Especially, as discussed in section 3.1.6, Andrew Lacey presented a simple probabilistic approach which yields a comparable result to those derived from complicated algorithms.

Through the previous works of Hatzivassiloglou and K. R. McKeown 1997 [13], Michael Gamon et al [17], and Andrew Lacey [20], we speculate that unsupervised learning may be quite appropriate to in-domain learning tasks of sentiment analysis. Therefore we implement our own unsupervised learning based on Andrew Lacey's method.

We implement the unsupervised learning algorithm using the WN-Similarity package. We calculate the sentiment orientation score for each selected phrase, and then let the sentiment score of a document equal to the average score of all phrases occurring in that document.

We will discuss the feature sets for unsupervised learning, the automatic SO score assessment for phrases, the word sense disambiguation problem, and the automatic SO score assessment for reviews in turn in the following sections.

4.3.1 Features for Unsupervised Learning

Our unsupervised learning algorithm is based on the similarity and relatedness provided by WN-Similarity. The idea behind the algorithm is that we assign a score of sentiment orientation to every phrase in an extracted training set; afterwards, each test document obtains a score equal to the average score of the features appearing in that document.

We implement this algorithm on Unigrams, Feature Set 1 and Feature Set 2. The distributions of these three feature sets were shown in Table 4.6:

POS	Unigrams				Feature Set 1				Feature Set 2			
	Total	Pos	Neg	Ntrl	Total	Pos	Neg	Ntrl	Total	Pos	Neg	Ntrl
noun	409	186	223	0	201	86	115	0	179	80	99	0
verb	462	203	259	0	255	120	135	0	261	135	126	0
adjective	550	215	335	0	312	133	179	0	330	145	185	0
adverb	26	15	11	0	25	12	13	0	23	10	13	0
bigrams					655				655			

Table 4.6 The distribution of 1447 features
in Feature Set 1 and Feature Set 2
(Pos:Positive; Neg:Negative; Ntrl: Neutral)

All the three feature sets comprise 1447 attributes in which there are 792 unigrams and 655 bigrams in both Feature Set 1 and Feature Set 2. The difference between them is that in Feature Set 1, all 792 unigrams are selected by their SO-PMI score, while in Feature Set 2, 396 unigrams with lowest SO-PMI scores are replaced by other 396 unigrams having highest SO_WN score.

Therefore, the 655 bigrams' sentiment score is equal to their SO-PMI score; whereas the score of the 792 unigrams will be calculated using the $SO(w)$ algorithm, which relies on WN-Similarity, as described in section 4.3.2.

The Unigrams feature set includes 1447 unigrams only. On the other hand, because Feture set 3 involves four synthetic features Average_PMI, Average_lesk, Average_hso, Average_jcn, it is unreasonable to directly assign sentiment scores to those synthetic features. Therefore we do not implement unsupervised learning on Feature Set 3.

Finally, we will compare the learning results on the three feature sets and investigate the effect of features selected by $SO(w)$ scores based on WordNet.

4.3.2 Automatic Sentiment Orientation Assessment for Phrases

Given k pairs of seed words for each part of speech, we assume that each pair contains a positive and a negative word. The positive seed word is denoted by *key-p*, and the negative seed word is referred to as *key-n*. We use $SO(w)$ to denote the sentiment orientation of word w, and setup the default threshold to zero; in other words, a $SO(w)$ value greater than zero means positive, otherwise negative. The absolute value of $SO(w)$ represents the strength/intensity of the sentiment orientation of word w.

The SO score of word w is described as following:

$$SO(w) = \sum_{i=1}^{k} Similarity(key - p_i, w) - \sum_{j=1}^{k} Similarity(key - n_j, w) \quad \textbf{(4.4)}$$

Similarly to what was shown in Table 4.3, we extracted 20 most frequent pairs of sentiment words for each part of speech including noun, verb, adjective and adverb. However, in our experiments, the k is set to 10, because there are two particular situations that we need to pay attention to, as described in section 4.2.4.3: First, because the WN-Similarity package returned an excessively high score when calculating the similarity between seed words and themselves, we have to avoid calculating the similarity between seed words themselves, i.e. in formula 4.4, we should not add the $Similarity(key - p_i, w)$ or $Similarity(key - n_j, w)$ when $key - p_i = w$ or $key - n_j = w$; Secondly, due to the disturbance resulting from relatedness between antonyms described in section 4.2.4.3, we use the $synset->antonyms() function of Lingua::Wordnet package to find the antonyms of w from the 20 seed words, and do not add their similarity with w when using formula 4.5 to calculate $SO(w)$.

Finally, starting with a given phrase/word w, we compute the similarity between w and 10 given pairs of seed words. From the given 20 seed words shown in Table 4.3, let *key-p* (or *key-n*)= $sw_1, sw_2, ..., sw_{10}$ be the list of 10 top-ranked positive (or negative) seed words sorted in decreasing order of their frequency. Of course, in these 10 top-ranked seed words, both the

word w itself and its antonyms have already been excluded to avoid the abnormally high *SO(w)* score which is mentioned in section 4.2.4.3. Thus, we replace the formula 4.4 with following formula 4.5:

$$SO(w) = \sum_{i=1}^{10} Similarity(key - p_i, w) - \sum_{j=1}^{10} Similarity(key - n_j, w) \quad \textbf{(4.5)}$$

As a result, both of the above disturbing factors are eliminated.

In addition, for the convenience of scoring reviews using the five star scheme, we divide $SO(w)$ by two to force the value of sentiment orientation to fall into [-5,+5]. This normalization step will be mentioned again in section 4.3.4.

4.3.3 Problems related to Word Sense Discrimination

We calculate $Similarity(key - p_i, w)$ and $Similarity(key - n_j, w)$ using the WN-Similarity package provided by S. Patwardhan and T. Pedersen's [30] [32]. WN-Similarity separately implements measures of similarity and relatedness that are all in some ways based on the structure and content of WordNet (see section 2.7 of Chapter 2). In the WN-Similarity package, there are three similarity measures and six relatedness measures. WordNet is particularly well suited for similarity measures, since it organizes nouns and verbs into hierarchies of 'is-a' relations. Is-a relations in WordNet do not cross part of speech boundaries, so the three similarity measures are limited to making judgments between noun pairs (e.g., *cat* and *dog*) and verb pairs (e.g., *run* and *walk*). Therefore, in formula 4.4, if w is a noun or a verb, its $SO(w)$ score is computed by the similarity measure *wup*; On the other hand, if the w is an adjective or an adverb, its $SO(w)$ value is calculated by the relatedness measure *jcn*.

First, because all the similarity measures of the WN-Similarity package is calculated based on

a pair of word senses in the WordNet hierarchy, the *Similarty(key ,w)* should also be drawn from a pair of given word senses. Actually, for each sense of the unigram w_i, we determine the similarity with each of the words in the list *key-p* or *key-n*, using a WN-Similarity measure of semantic similarity. However, either the unigram w_i or seed words *key-*p_i/*key-*n_j are themselves ambiguous, so there are many possible combinations of w_i and *key-*p_i/*key-*n_j.

This thesis does not delve deeply into the problem of word sense disambiguation, but we have to choose a strategy for determining the word sense used to calculate the similarity. There are two strategies:

1. When calculating the similarity between w_i and *key-*p_i/*key-*n_j, for all word senses w_i and all word senses *key-*p_i/*key-*n_j, we use the sense that maximizes the similarity score to denote the *similarity(key-*p_i,*w)*/ *similarity(key-*n_j,*w)*

2. A selection process can be applied so that an ambiguous seed word belongs only to one sense. In this case, for a given sense w_i we use only those seed words with whom w_i has the highest similarity score across all the senses of *w*.

Based on the 10 most frequent seed words from Table 4.2, we calculate the *similarity(key-*p_i,*w)*/ *similarity(key-*n_j,*w)*. According to our investigation and observation, the second strategy forces the seed words belonging to one preselected sense to significantly improve the result of scoring the similarity. Lastly, we only use the second strategy in our experiments. As discussed in section 4.2.4.2, in this situation, we adopt the first sense strategy that uses the first sense of words as its default sense.

Secondly, all the features present the spectrum shown in Table 4.7:

Word	***Score***
horrendous	1.2081
dishonest	1.2702
illegal	1.3548
refuse	1.4118
ludicrous	1.5454
...	...

accept	3.2857
free	3.7631
...	...
pleased	4.3333
navigable	4.5
great	4.6170

Table 4.7 Sentiment Terms in Feature Set 1

Intuitively, the words at each end of the list should be included, while those in middle should be eliminated because they might not be strong sentiment words. However, after a simple test, we found that the phrases in the middle are indispensable. First, all the 1447 features already distribute very sparsely in all 3164 reviews, so sparseness will become a more serious problem if the feature set shrinks further; Secondly, the words located in the middle are actually close to neutral sentiment orientation, and contribute a lot to finely adjust the scoring of a whole review; In other words, they are representative when measuring the sentiment of a review. On the other hand, deleting these features negatively influences the algorithm accuracy. Therefore, we do not ignore these 'moderate' sentiment terms.

4.3.4 Automatic Assessment of Reviews and Results

By using *wup* and *jcn* measures of WN-Similarity package, the function *Similarity*(*key*,*w*) always return a real value in the scope of (0,1]. Theoretically, for any word, the *SO*(*w*) falls into the closed interval [-10,10]. For the following prediction, we need normalize the score of phrases from [-10,10] by dividing the initial *SO*(*w*) value by two, so the *SO*(*w*) finally falls into [-5,5].

With the normalized *SO*(*w*), we assign *S*(*d*), the sentiment score of review, by the method from Lacey [20]:

$$S(d) = \frac{\sum [SO(w) \mid (w \in d) \wedge (w \in L)]}{|[W : (w \in d) \wedge (w \in L)]|} \qquad \textbf{(4.6)}$$

in which, L means the list of features, d denotes the document which the w belongs to, w is a feature word or a phrase, W means the set of features appearing in both the current review and the feature set L which is currently used.

Generally, in customer reviews, '*poor*' corresponds to score 1 (one star) and '*excellent*' represents score 5 (five star), so there is no score of '0' in the five star system. This scoring scheme was shown in Figure 1.1. Intuitively, we can treat score 3 (good) as a middle score when assigning score to a review. However, because the absolute value of sentiment score $SO(w)$ is between 0 and 5, the value of $S(d)$ will be between [-5,+5]. For using the $S(d)$ to score reviews by the five star scheme, we need to map $S(d)$ from interval [-5,+5] to [-2,+2], and then add it to the middle score 3. Therefore, if we let $Score(d)$ be the final five star score of document d, we have:

$$Score(d) = round(3 + \frac{2}{5} \times \frac{(S(d) - S(d)_{avg})}{(S(d)_{max} - S(d)_{min})}) \qquad \textbf{(4.7)}$$

In which, the $S(d)$ is the sentiment score figured out by formula 4.7, $S(d)_{avg}$ is the average score of all 3164 reviews, $S(d)_{max}$ is the maximum value and $S(d)_{min}$ is the minimum value in all reviews. Finally, the result is rounded to 3 decimal places, that is the unsupervised scoring result $Score(d)$. In our experiments, the $S(d)_{max}$ is 4.698, the $S(d)_{min}$ is 1.034, and the $S(d)_{avg}$ is 2.637.

We implemented the algorithm on Unigrams, Feature Set1 and Feature Set 2, to compare the effects without, before and after incorporating SO_WN measure with the feature set. The detail of experimental results and analysis about unsupervised learning will be discussed in section 5.2.

4.4 Supervised learning

It is natural to think of sentiment analysis as a multi-class classification problem. From the

reviews from Epinions [27] (http://www.epinions.com), we can extract five star rating information. Thus, we can think of the five star rating as a five-class classification problem to categorize reviews into classes of 1, 2, 3, 4, and 5 as shown in Figure 1.1 (Note: There is no score of zero). In this section, we use the high-frequency sentiment phrases and additional attributes as features, and execute four supervised learning algorithms on four different feature sets and compare their performance.

The four supervised learning algorithms used in supervised learning are Naïve Bayes, BayesNet, C4.5 decision tree, and Support Vector Machine (SMO in WEKA). The four feature sets are unigrams, Feature Set 1, Feature Set 2 and Feature Set 3 as described in section 4.2.

4.4.1 The selection of algorithms

According to our preliminary study in chapter 3, in many previous works, SVM outperformed other machine learning algorithms for sentiment classification tasks; for example, Bo Pang et al [10], Michael Gamon and Anthony Aue [17], Tony Mullen and Nigel Collier [18] and Alistair Kennedy and Diana Inkpen [11] etc. Hence, SVM is an important algorithm we need to examine in this thesis.

Moreover, many previous works presented found that very weakly supervised methods for sentiment classification are especially desirable and promising choices. For example, in Bo Pang et al [10] (discussed in 3.1.7), Naïve Bayes presented comparable performance to that of more sophisticated algorithms such as SVM. In addition, in the paper by Michael Gamon & Anthony Aue 2005 [17] (discussed in 3.1.5), they obtained good performance by combining Naïve Bayes with the bootstrapping approach for indentifying sentiment vocabulary.

We think this phenomenon is related to the highly domain-specific nature of sentiment analysis: due to the specialty of each domain of content, the simplicity of methodology of the simple classification approaches such as Naïve Bayes and BayesNet are more beneficial and effective for modeling in-domain machine learning tasks than complicated algorithms.

Therefore, we also choose these simple but effective classifiers, i.e. Naïve Bayes and BayesNet, in our supervised learning.

Lastly, as a traditional learning algorithm, C4.5 decision tree plays a very important role in text categorization. For example, in Evgeniy Gabrilovich et al [73], the performance of C4.5 was competitive with improved SVM. Hence, we also use C4.5 in our supervised learning experiments, compare its performance with that of the other three algorithms including Naïve Bayes, BayesNet and SMO, and observe its accuracy on different feature sets.

Generally, the probabilistic methods are quantitative (i.e., numeric) in nature, and as such have sometimes been criticized since, effective as they may be, they are not easily interpretable by humans. For example, the Naïve Bayes algorithm is based on probabilistic method; it is usually effective for numeric prediction while it is not easy to understand. However, symbolic algorithms, such as decision tree learners, do not suffer from this problembecause they are inductive rule learners.

Actually, our multiclass classification is a type of nominal prediction (before we make use of the order information among all five star score). Therefore, intuitively C4.5 should surpass both Naïve Bayes and Bayes Net; and we speculate that C4.5 can achieve a competitive performance with SMO. In fact, the result in our experiment goes contrary to our expectation, and SVM outperforms C4.5. The experiments and results will be discussed in detail in chapter 5.

To sum up, our goal is to check how various classification algorithms perform on differrent feature sets, improve their learning performance, analyze the result, and explore the possibilities of further improvement at present or in the future.

4.4.2 Motivation and Heuristics

For supervised learning, we treat the five-star rating task as a five class classification problem.

We have no intention to traverse that ground over all the possible combinations thorough all different classifiers and different feature sets. In the process of our experiments, we gradually eliminated some ineffective algorithms and feature sets through the comparation among them. Once we found that the performance of these algorithms or feature sets were obviously surpassed by other algorithms and feature sets., This method helped to reduce our experimental workload so that we could focus on the more promising solutions and avoide hopeless and useless expense.

Consequently, we run Naïve Bayes, BayesNet, C4.5 and SMO on each feature set from unigrams, Feature Set 1, Feature Set 2, and Feature Set 3. All the experiments are implemented in WEKA version 3.5.6. Beginning with all four classifiers, we exclude the low-performance classifiers in turn from the four optional classifiers, gradually narrowing down the scope of candidate classifiers, and focus our exploration to those that achieve higher accuracies. Similarly, we may delete a feature set from the testbed of our experiments, if we find all classifiers perform obviously and consistently bad.

In Bo Pang et al [10], they concluded that when using Naïve Bayes, Maximun Entropy and SVM to perform sentiment analysis for movie reviews, feature presence achieved much better performance than feature frequency.

However, in this thesis, the scenario is different because we are quantitatively predicting the sentiment score, so we hypothesize that frequency should be weighted more than presence for expressing the extent sentiment orientation. Therefore, we also experiment with supervised learning algorithms on different weighting methods, as shown in Table 5.3, including (all results are displyed in Table 5.3):

1. presence
2. frequency
3. tf/idf
4. frequency*$SO(w)$

5. presence_POS
6. frequency_POS

As shown above, in addition to the familiar attributes 'presence', 'frequency', and 'tf/idf', we also experiment with 'frequency*$SO(w)$', 'presence_POS', and 'frequency_POS'.

Because both frequency and the sentiment orientation score positively affect the subjectivity of reviews, we multiply them to get the 'frequency*$SO(w)$' and explore whether this weighting method performs well.

This idea is derived from the concept of gradation of adjectives. As discussed in section 3.1.2, according to the study by Hatzivassiloglou and J.M.Wiebe [16], gradability has the ability to intensify or diminish the modified noun. Moreover, Lyons [34] also argued that gradability, instead of truth/false functions, provides the better explanation of these differences between sentiment features. We speculate that the sentiment orientation score $SO(w)$ may act like a good gradability indicator.

On the other hand, intuitively, because we think repeated sentiment words seem likely to intensify the degree of favorability, we would like to use the frequency as another factor of the value of the feature. Therefore we use 'frequency of word(w_i) * $SO(w_i)$' as the value of attributes, i.e. the fourth item shown above, and observe its effect.

Furthermore, Bo Pang et al [10] had experimented with appending POS tags to every word to take advantage of word sense disambiguation. As discussed in section 3.1.7 (as shown on line 5 in Table 3.6), the accuracy improved for Naïve Bayes, but declined for SVM. They did not confirm this effect in their conclusion, but we are interested in this additional information about POS. Therefore we also append POS disambiguation into the attributes and explore whether it can benefit to these four classifiers.

4.4.3 Experimental Set-up

So far, we have four feature sets: the unigram feature set comprises 1447 unigrams, both feature set 1 and feature set 2 include 792 unigrams and 655 bigrams, while Feature Set 3 is extended by adding four features of Average_PMI, Average_lesk, Average_hso, Average_jcn (see section 4.2.5.2) based on Feature Set 2. All four feature sets use the vector space model, which is one of the most widely used models in Natural Language Processing (hereafter NLP), using spatial proximity for semantic proximity. As a result, all customer reviews are represented in a high-dimensional space, in which each dimension of the space corresponds to a feature in our feature set. Feature set 2 is used to explore, compared to feature set 1, how WordNet selected features perform by supervised learning; while feature set 3 is used to determine the effectiveness of the four WordNet derived average synthetic information measures.

Before starting these learning processes, it is emphasizing the importance of the unigram feature set. According to the preliminary study in Chapter 3, in both Tony Mullen et al [18] and Bo Pang et al [10], the unigram feature set performs the best or second best in supervised learning. Therefore, we also create the unigram feature set of 1447 unigrams, to compare its performance with hybrid feature sets comprised of unigrams and bigrams such as Feature Set1, Feature Set 2, and Feature Set 3.

Furthermore, in the top 1447 most frequent unigrams, there are some words occurring more than once with different senses, so we also experiment with appending POS tags to these 1447 unigrams (become 1501 unigrams). All the POS tags are extracted from lemmatized reviews by GPoSTTL tagger.

In addition, we experiment with supervised learning on the frequency of sentiment phrases too due to the difference between the binary classification of favorability and multiclass classification of five star scoring. Actually, to some extent more occurrences of a sentiment term should weight more in sentiment scoring of a review than its presences should. Similarly, we think about other possible term weighting methods including *tf/idf* and the frequency by a

factor of *SO(w)*. To sum up, as mentioned in section 4.4.2, there are six weighting strategies shown as follows:

1. Presence

   The value of the feature only reflects the presence or absence of a feature. We assign 1 to the attribute if the phrase appears in the review, otherwise we assign 0.

2. Frequency

   The information captured by term frequency is how salient a word is within a given document. The higher the term frequency the more likely it is that the word is a good description of the content of the document.

3. tf/idf

   The frequency of a word indicate higher importance, but not as much as the count would suggest, so the term frequency is usually dampened by a function like $f(tf) = \sqrt{tf}$ or $f(tf) = 1 + \log(tf), tf > 0$, as shown in the third line of Table 4.8. On the other hand, the document frequency can be interpreted as an indicator of informativeness.

   One way to combine the phrases' term frequency and their document frequency into a single weight is as following formula (4.8):

$$weight(i,j) = \begin{cases} (1 + \log(tf_{i,j})) \log \frac{N}{df_i} & if\ tf_{i,j} \geq 1 \\ 0 & if\ tf_{i,j} = 0 \end{cases} \quad \textbf{(4.8)}$$

   This form of document frequency weighting is often called inverse document frequency or *idf* weighting. More generally, the weighting scheme in (4.8) is so-called *tf/idf* weighting scheme.

For getting the inverse document frequency, there are different options when thinking about term occurrence and document frequency. They are listed below in the second column of Table 4.8:

Term Frequency	tf/idf		Weighting by SO(w)
	Term occurrence	Document frequency	
Frequency √	n (natural) $tf_{t,d}$	n (natural) df_t	*frequency*SO(w)* √
	l (logarithm) $1 + log(tf_{t,d})$	*t* *log(N/df$_t$)* √	
	a (augmented) $0.5 + \frac{0.5*tf_{t,d}}{max_t(tf_{t,d})}$		

Table 4.8 Components of tf/idf weighting schemes. *tft,d* is the frequency of term *t* in document *d*, df_t is the number of documents *t* occurs in, *N* is the total number of documents, and w_i is the weight of term *i*.

tf/idf is a necessary means to deduct the unreasonable importance of the undampened count of highfrequency words, and different combinations of term occurrence and document frequency can be applied to different tasks in NLP. In our experiment, we adopt 'lt' combination (the second line of 'l' and 't') in Table 4.8 as optional features for the multiclass classification problem in next section

4. frequency*SO(w)

Because both frequency and the sentiment orientation of terms are important factors that positively related with the subjectivity of reviews, we multiply them and explore whether this weighting method perform well.

5. presence_POS

Some of the 1447 unigram features may be words with different senses. To perform crude word sense disambiguation, we attach the POS tag to the words, and treat each combination of word and POS as a separate word to count the presence.

6. frequency_POS

As above, we attach the POS tag to words and count the frequency for each combination of word and POS.

4.5 Encountered problems and corresponding solutions

4.5.1 General Problems

In order to increase learning performance, we intend to improve each aspect of our sentiment analysis task. In addition to the feature selection and knowledge representation mentioned above, we also adopt the methods of engineering input and output to improve the results of the machine learning algorithms used in this thesis.

As discussed above, we have already adopted some different ways to make the input more amenable:

a. Using large dataset

   We expand our dataset from 618 reviews of only two banks to 3164 reviews of 46 banks.

b. Using cross-validation

   We use 10-fold cross-validation for supervised learning.

c. Engineering the input data into a format suitable for learning algorithms

   We try using different weighting methods in attribute representation, as discussed in section 4.4.3.

d. Adding new synthetic attributes,

   As metioned in section 4.2.3, we make use of WordNet score SO_WN and synthetic measures including Average_PMI, Average_lesk, Average_hso, Average_jcn in the Feature Set 3.

However, going with the development of our experiments, we find that there still is a lot of room for improvement of the input and output phases:

a. Filtering the input in different ways
b. Using the ordinal information hiding in five star scoring scheme
c. Combining different models and techniques learned from the bank reviews
d. Using effective techniques for improving the phase of output, such as error-correcting code, boot-straping, stacking and so on.
e. Dealing with special problems related to a particular dataset, for example, class imbalance.

In this thesis, we specially address the issues of utilizing the ordinal information of class attributes and solving the problem of imbalanced data. We discuss these issues and related solutions respectively in section 4.5.2 and 4.5.3.

4.5.2 Utilizing the ordinal information contained in class attribute

When using machine learning to solve five star scoring problems, the score of reviews exhibits an order among the so-called class attributes. So far, however, the classification algorithms shown in section 4.4 do not use this ordering information because they all treat score of reviews as a nominal quantity.

We use WEKA as the learning tool in our experiments. There are four types of measurements: nominal, ordinal, interval and ratio quantities. According to Frank Eibe [59], the ordinal

measure is different from nominal because the former exhibits an order among attribute values while the later does not. Interval values exhibit an order too, but it is more constrained since they are measured in fixed and identical units. The five star scoring seems similar to the interval value, but it is only an approximation to the interval value, because the distance between any two adjoining scores, such as the difference between score 2 and 3 is not necessarily always equal to the difference between another pair of adjoining scores, such as the distance between score 4 and 5.

In this thesis, we speculate that multi-class classfiers can produce relatively better experimental results than that of unsupervised learning. Unfortunately, standard multi-class classification ignores the ordering information hiding in the class attribute, although this information could improve the performance of classifier. Therefore, we utilize the method proposed by Eibe Frank [59] to exploit ordinal information from score 1 to the score 5.

Figure 4.6 shows the process of how to incorporate ordinal information into the standard classification learner. First, the vectors of reviews are transformed from five class ordinal problem to four binary class problems. We divide ordered scores from 1 to 5 into 4 binary classification datasets. The first has a class attribute that represents *score*>1, the second has a class attribute representing *score*>2, and in turn, the third and fourth represents *score*>3 and *score*>4. Each derived dataset contains the identical number of attributes as the original dataset, with the same attribute values for each instance except the class attribute. The class attribute just is the Boolean value of *score*>1, *score*>2, *score*>3 or *score*>4.

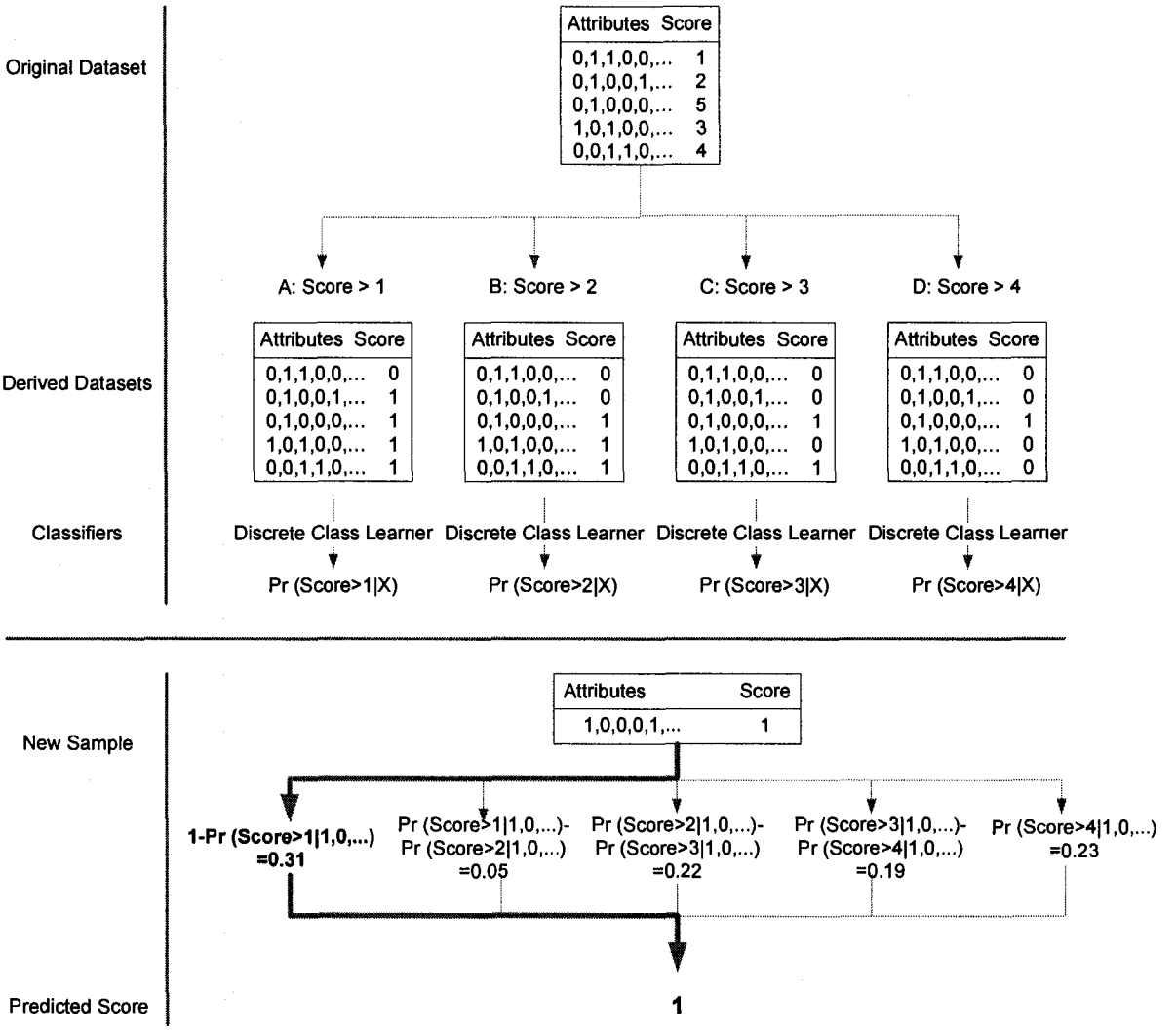

Fig 4.6 Dividing multiclass problem into four binary class problems

When predicting the class of new instances, we need the probabilities of the classes using the *k-1* models. The estimation of the probability for the first and last ordinal class value depends on a single classifier. The probability of the first ordinal value 1 is given by 1-*Pr*(*Score*>1),while the last ordinal value 5 is calculated from (*Score*>4). The probability of class value of 2, 3 and 4 depends on a pair of classifiers. They are given as follows:

$$Pr(2)=Pr(Score>1)-Pr(Score>2)$$
$$Pr(3)=Pr(Score>2)-Pr(Score>3)$$
$$Pr(4)=Pr(Score>3)-Pr(Score>4)$$

In WEKA, we get the learning result of binary classification with probability as shown in

Figure 4.7:

inst#,	actual,	predicted,	error,	probability distribution				
1	5:5.0	5:5.0		0.005	0.01	0.039	0.24	*0.706
2	5:5.0	5:5.0		0.002	0.01	0.037	0.226	*0.725
3	5:5.0	5:5.0		0.005	0.012	0.023	0.197	*0.763
4	5:5.0	5:5.0		0.075	0.072	0.062	0.223	*0.567
5	5:5.0	1:1.0	+	*0.624	0.204	0.099	0.03	0.044

Fig 4.7 The probability distribution of binary classification in WEKA

Therefore, for new instances in test sets, the classification is processed with each of the four classifiers and the probability of each of the 5 ordinal class values is computed using the above method shown in figure 4.6. Finally, the class with maximum probability is assigned to the instance.

4.5.3 Solving the problem of imbalanced data

In supervised learning experiments, we treat the five-star rating task as a five class classification problem. The 3164 bank reviews extracted from www.epinions.com do not present a uniform class distribution.

	Score	Number of Instances
Class1	1.0	1436
Class2	2.0	442
Class3	3.0	268
Class4	4.0	408
Class5	5.0	610
Skew Value		+1.868

Table 4.9 The distribution of instances of 3164 reviews

As shown in Table 4.9, all the samples are not equally distributed through five classes. In this thesis, we are not focusing on the problem related to imbalanced data, but our study to some

extent inevitably overlaps with the scope of skewed class classification problem. As shown in Table 4.9, the skewness of the distribution of bank review instances is +1.868. This value means that there is a relatively large degree of asymmetry, and biases the classification toward the majority class, i.e. class 1. This characteristic negatively affects the learning result, so we address this problem and explore the solutions of re-sampling and combining different learning models respectively in section 4.5.3.1 and 4.5.3.2.

4.5.3.1 Re-sampling

As mentioned above, when there are many more instances of some classes than others in a dataset, we consider this dataset to be imbalanced. Generally, when learning from datasets with imbalanced class distributions, machine learning algorithms tend to produce biased classifiers because standard classifiers tend to be overwhelmed by the large classes and ignore the small ones.

Table 4.10 shows that in our dataset of customer reviews, the largest class of score 1 has 1436 instances which are almost equal to the total of instances in class 2 (442 instances), class 4 (408 instances) and class 5 (610 instances), and more than 5 times of instances of class 3 (268 instances). Obviously, the degree of the imbalance problem is quite considerable, and the bias toward the majority is relatively serious.

	Score	Number of Instances	TP Rate	FP Rate	Precision
Class1	1.0	1436	0.909	0.388	0.661
Class2	2.0	442	0.023	0.017	0.182
Class3	3.0	268	0.112	0.027	0.275
Class4	4.0	408	0.201	0.061	0.328
Class5	5.0	610	0.669	0.144	0.526

Table 4.10 Learning Result of BayesNet on 1447 unigrams

Corresponding with the imbalanced class distribution, the learning result, such as the precision, on the largest class of score 1 outperformed the result of each among the precisions of all other classes, and especially, far better than the result based on the smallest class of score 3. Figure

4.8 shows the confusion matrix of the BayesNet algorithm on 1447 unigram features using the word presence weighting method.

```
=== Confusion Matrix ===

   a    b    c    d    e   <-- classified as
1305   16   30   27   58 |    a = 1.0
 336   10   18   23   55 |    b = 2.0
 128   10   30   41   59 |    c = 3.0
 101   10   20   82  195 |    d = 4.0
 105    9   11   77  408 |    e = 5.0
```

Fig 4.8 Confusion Matrix of BayesNet Algorithm

This result is consistent with the study by Weiss et al [60], in which the experimental evidence shows that examples belonging to the minority class are misclassified more often than examples belonging to the majority class. Their experiments implied that classifiers tend to perform worse on the minority class than on the majority class, similar to the learning performance of class 3 and class 1 in our experiments.

Generally speaking, there are two kinds of approaches addressing the imbalanced class distribution problem. One is using re-sampling methods to change the class distribution or misclassification costs of original dataset; another one is adjusting classifiers to the imbalanced datasets. We explore the re-sampling methods in this section. In next section, we will try to combine the score-prediction method of unsupervised learning and five-class classification meta-classifier to deal with the problem of skewed dataset.

Many highly imbalanced problems have non-uniform error costs that favor the minority class, e.g., some environmental problems such as nuclear radiation detection. In these kinds of learning problems, more attention needs to be paid to the errors on the minority class due to their higher costs and risks. However, in our five star scoring problem, we consider all classification errors to be equally important because there is no difference in the importance of the different classes. Therefore, our goal is to compensate for the negative impact resulting from the lack of samples of given classes and improve the accuracy of classification. In other

words, we are focusing on increasing the number of minority samples.

We adopt randomly generated subsamples of our training dataset without replacement, and do not bias the classification toward a uniform distribution. We use only 1 seed for subsampling and set the new subsample size equal to the original dataset. Obviously, this configuration tends to minimize the overfitting during the re-sampling and avoid forcibly creating a uniform class distribution.

In addition, according to the research of [61][62], the cost-sensative method does not really outperform the simple re-sample methods in imbalanced class problems, so we do not modify the default cost of errors in every class.

4.5.3.2 Combining Supervised Learning and Unsupervised Learning

Meta-learning inspires us with the idea of combining different learning methods to improve the learning performance. Coincidently, we found that the imbalanced distribution of instances holds on almost all of 46 banks from which we extracted reviews. In other words, this imbalanced data distribution is a regularity instead of a mere coincidence for reviews of banks. In this thesis we do not investigate the cause of the imbalanced data distribution of bank reviews, and we only focus on how to deal with this special phenomenon.

The unsupervised learning results on Feature Set 1 and Feature Set 2 are shown in Table 4.11. The precisions of different classes indicate that unsupervised learning is good at scoring reviews in minority classes 2, 3, and 4. We incorporate the results of the meta-learning algorithm based on BayesNet (with re-sampling) into Table 5.1 and create table 4.11, for comparing the performance of both methods and investigating their respective characteristics.

		Unsupervised Learning						**Meta Learning based on BayesNet**					
	Class	**1.0**	**2.0**	**3.0**	**4.0**	**5.0**	**Precision**	**1.0**	**2.0**	**3.0**	**4.0**	**5.0**	**Precision**
Featur	**1.0**	1191	79	155	11	0	0.735	1041	175	93	61	66	0.746
	2.0	234	44	151	13	0	**0.232**	218	67	46	45	66	**0.188**
	3.0	51	28	157	30	2	**0.224**	61	40	48	52	67	**0.188**

	4.0	15	9	197	171	190	**0.499**	31	44	28	115	190	**0.294**
	5.0	45	30	40	118	42	0.179	45	30	40	118	377	0.492
Feature Set 2	**1.0**	1191	79	148	18	0	0.748	836	326	139	46	89	0.755
	2.0	246	44	134	30	0	**0.265**	162	121	69	36	54	**0.197**
	3.0	51	28	125	62	2	**0.185**	43	62	56	45	62	**0.145**
	4.0	15	9	137	216	30	**0.302**	37	54	64	83	170	**0.259**
	5.0	8	6	132	389	74	0.698	29	52	57	110	362	0.491

Table 4.11 Comparison of matrices and learning precision between unsupervised lerning and meta-learning

The metrices and precisions presented in Table 4.11 are calculated on Feature Set 1 and Feature Set 2. (The unsupervised learning was only performed on these two feature sets) As can be seen from line 2 to line 4 of classes 2, 3, and 4, the precision of the unsupervised learning are higher than that of the meta-learning. The same situation appears from the line 7 to line 9.

However, as shown in line 1, 5, 6 and 10 of the Table 4.11, meta-learning outperformed unsupervised learning on class 1 and class 5 towards which the meta-classifier is biased.

Our unsupervised learning method is developed based on Andrew Lacey's approach [20]. This unsupervised method focuses on grammatical structures and assigns the sentiment score to terms and phrases, and then uses scores of these sentiment terms to evaluate the sentiment score of reviews of the test dataset. Indeed, the unsupervised learning is very sensitive to capture the sentiment contained in natural language and not apt to be affected by the skewed datasets. However, unsupervised learning does not take advantage of the quantity of the samples in majority classes, as supervised learning does.

On the other hand, it is hard to get rid of excessively biasing toward majority classes with supervised learning even though re-sampling to some extent can compensate for the negative effect of imbalanced data. Thus, it makes sense that we combine the score prediction results of classes 2, 3 and 4 of unsupervised learning and the prediction results of classes 1 and 5 of supervised learning. We speculate that this combination can benefit from the strong points of both methods and offset their weaknesses. The combination procedure is shown in Figure 4.9:

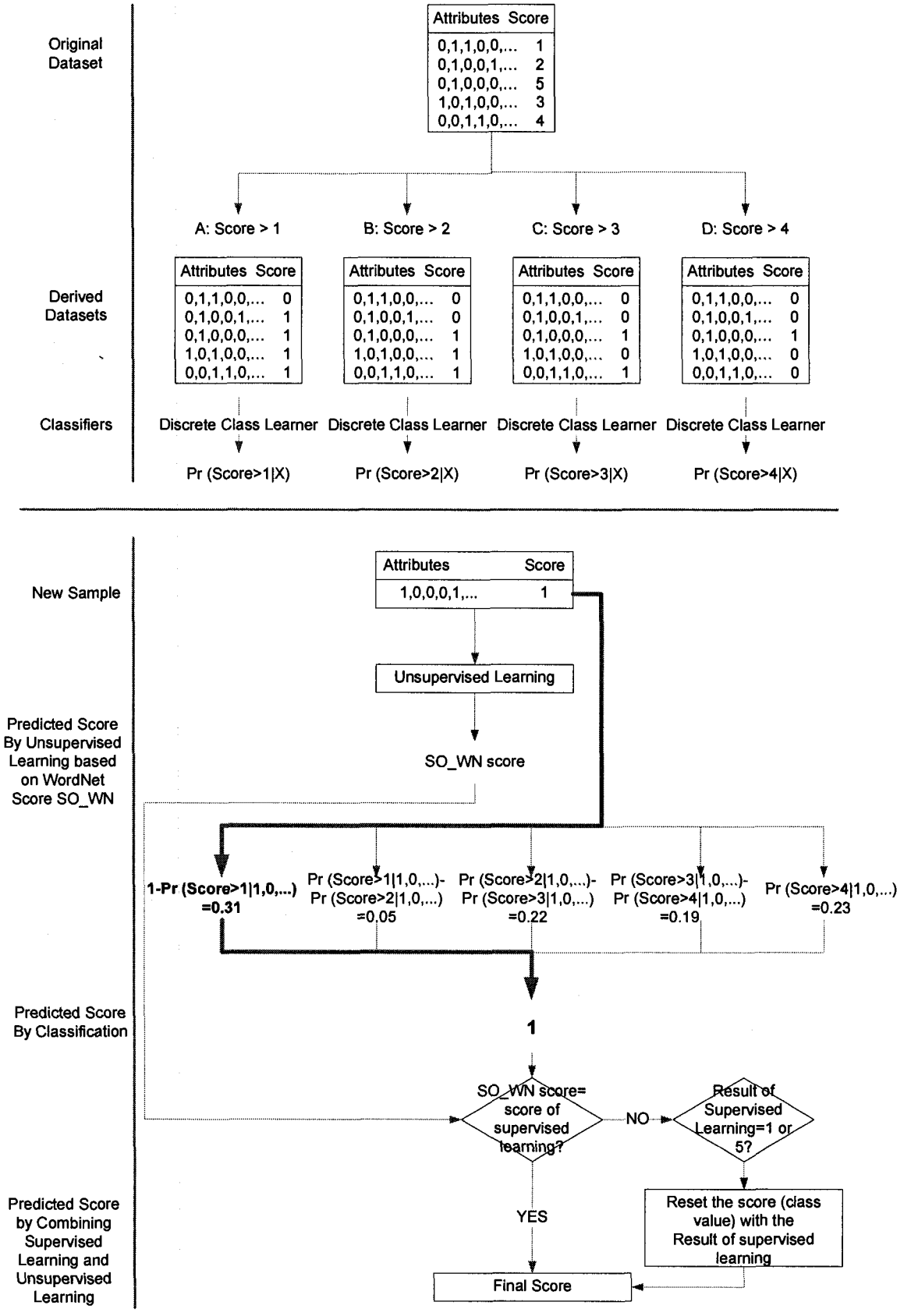

Fig 4.9 Combining Unsupervised learning and Supervised learning

Chapter Five
Experiments and Results

5.1 Corpus Preprocessing and Feature Selection

In order to extract learning features and calculate $SO(w)$ from the review corpus, we used the following steps:

- Gather the 3164 reviews of 46 banks from Epinions (www.epinions.com). All 46 selected banks have no less than ten customer reviews. Appendix A shows the names of the banks and the number of reviews per bank
- Transform the HTML files of reviews into plain text format
- Lemmatize the corpus and tag the lemmatized words with their Part of Speech (POS) using the GPoSTTL tagger
- Strip the stop words with two stop word lists, and filter out proper nouns
- Select the sentiment nouns, verbs, adjectives and adverbs which are contained in the General Inquirer (GI) and marked as 'Positiv' or 'Negativ' in the 'Positiv' and 'Negativ' fields in GI.
- Use Turney's SO-PMI algorithm to expand sentiment terms. This algorithm uses SO-PMI scores to find new positive and negative lemmas or phrases. Then, the sentiment terms by GI and by SO-PMI algorithm are combined together to form Feature Set 1.
- Use WordNet to find novel sentiment lemmas outside of Feature Set 1. We use the *'jcn'* and *'lesk'* algorithms provided by the WordNet::Similarity package to calculate the relatedness score between candidate lemmas and preselected seed words. All lemmas whose relatedness scores are higher than the threshold become new sentiment terms. Replacing some terms of Feature Set 1 with these new terms selected by WordNet relatedness, we build Feature Set 2.
- Add topical synthetic features of Average_PMI, Average_lesk, Average_hso, Average_jcn into the original feature set which includes only lemmas and bigrams, thus, obtaining Feature set 3.

Feature Set 1 and Feature Set 2 are used in unsupervised learning, while Feature Set 1 through Feature Set 3 and the unigram dataset are all used in supervised classification.

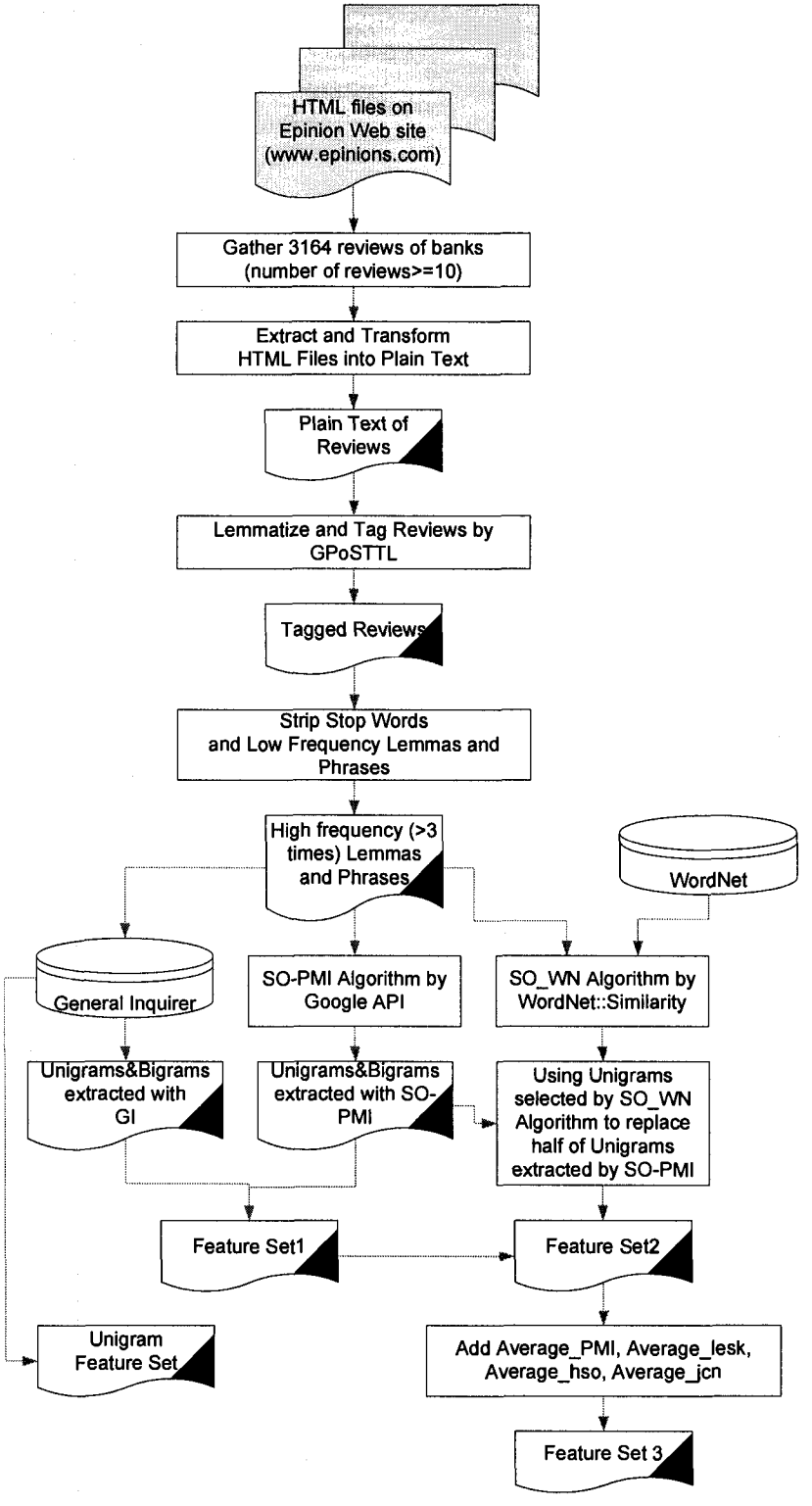

Fig 5.1 Flowchart of Preprocessing Process for Feature Selection

Figure 5.1 shows the flowchart of the preprocessing process for unigram feature set, Feature Set1, Feature Set 2 and Feature Set 3.

5.2 Unsupervised Learning

As discussed in section 4.3.4, we implemented the unsupervised learning algorithm on Unigrams, Feature Set1 and Feature Set 2, and obtained the final result presented in Table 5.1:

		Condusion Matrix					Error by class			Overall Error			
		1.0	2.0	3.0	4.0	5.0	TP	FP	Precision	CCI	ICI	MAE	RMSE
Unigrams	1.0	**1185**	81	157	13	0	0.823	0.191	0.729	1580	1584	0.3162	0.4327
	2.0	239	**39**	149	11	0	0.087	0.042	0.223				
	3.0	53	36	**129**	45	5	0.48	0.195	0.181				
	4.0	18	13	203	**186**	188	0.303	0.071	0.322				
	5.0	43	32	39	120	**41**	0.148	0.014	0.602				
Feature Set 1	1.0	**1191**	79	155	11	0	0.829	0.2	0.735	1605	1559	0.3023	0.41
	2.0	234	**44**	151	13	0	0.10	0.054	0.232				
	3.0	51	28	**157**	30	2	0.759	0.231	0.224				
	4.0	15	9	197	**171**	190	0.294	0.067	0.499				
	5.0	45	30	40	118	**42**	0.153	0.015	0.179				
Feature Set 2	1.0	**1191**	79	148	18	0	0.829	0.185	0.748	1650	1514	0.2282	0.3812
	2.0	246	**44**	134	30	0	0.097	0.045	0.265				
	3.0	51	28	**125**	62	2	0.466	0.190	0.185				
	4.0	15	9	137	**216**	30	0.531	0.181	0.302				
	5.0	8	6	132	389	**74**	0.122	0.013	0.698				

Table 5.1 Unsupervised Learning Result on Unigrams, Feature Set 1 & Feature Set 2 (CCI: Correctly Classified Instances ICI: Incorrectly Classified Instances MAE: Mean absolute error) (RMSE: Root mean squared error)

There are two clear characteristics of the unsupervised learning method: First, the distribution of the precisions on both feature sets is similar in the regularity that The score of 3 has the lowest precision in both Feature Set 1 and Feature Set 2, and the score 1 and score 5 have the highest precision values (the precision of score 5 in Feature Set 2 probably is an exception).

However, the skewed distribution of precisions among the five classes of unsupervised learning is not as serious as that of supervised learning as will be shown in Figure 5.6 and discussed in section 5.3.3.2. Obviously, the correctely classified instances of class 3 in unsupervised learning are more than that of supervised learning presented in Figure 5.6. In other words, the unsupervised learning is less affected by the skewed dataset than supervised learning is.

Secondly, compared to the random assignment method of five star scoring, the unsupervised learning achieves an acceptable performance. For a five star rating, the random-assignment baseline result would be around 20%. In our experiments using such a baseline, on Feature Set 1 and Feature Set 2, we obtain 20.6384% and 20.8281% scoring accuracies respectively as shown in Table 5.2. On the other hand, as also presented in Table 5.2, our unsupervised learning method achieves 50.7269% and 52.5215% accuracies respectively which outperforms our random-assignment baseline on Feature Set 1 and Feature Set 2.

As shown in Table 5.2, the accuracies – the percentage of documents scored correctly – of unsupervised learning for Feature Set 1 and Feature Set 2 outperform the randomly scored results. In addition, the Feature Set 2, which substitutes 396 WordNet-selected unigrams for 396 SO-PMI selected features, raises the accuracy to 52.52% from 50.73%. In other words, this result indicates that the features selected by our SO_WN contribute an obvious improvement for unsupervised learning.

Furthermore, as a whole, the accuracy of unsupervised learning on Unigrams is lower than both on Feature Set 1 and Feature Set 2. Although the performance on Feture Set 1 and 2 does not surpass that on Unigrams very much, we can conclude that in the five star rating task, only unigrams do not outperform combined unigrams and bigram phrases. This result is different from that of Bo Pang et al. [10], but the difference is not very significant. As shown in Table 5.2, the accuracy on Feature Set 1 is 0.7929% higher than that on Unigrams, and the accuracy on Feature Set 2 is 2.5848% higher than that on Unigrams. We can conclude that Unigrams do not beat Feature Set 1 and Feature Set 2 in unsupervised learning experiments.

	Correctly Classified Instances	Incorrectly Classified Instances	Accuracy	Error Rate
Unigrams	1580	1584	49.9367%	50.0632%
Random assignment for Unigrams	655	2509	20.7016%	79.2984%
Feature Set 1	1605	1559	50.7269%	49.2371%
Random assignment baseline 1	653	2511	20.6384%	79.3616%
Feature Set 2	1650	1514	**52.5215%**	47.4785%
Random assignment baseline 2	659	2505	20.8281%	79.1719%

Table 5.2 Results of unsupervised learning

The unsupervised learning also provides us with baselines for experimental comparison; in particular, the third baseline of 52.5215% might actually be considered somewhat difficult to beat, since some later experiments using the same feature sets as features for supervised machine learning methods do not necessarily yield better results.

Although we do not claim that our unsupervised scoring achieves the best accuracy, or that feature set 2 is the optimal set (because we speculate that a bigger proportion of WordNet selected features may produce higher accuracy), up to now, this result still is a good reference for multiclass classification methods.

5.3 Supervised Learning

5.3.1 Experimental Results

5.3.1.1 Learning Results

Based on the feature sets introduced in section 4.4.3, we implement BayesNet, NaiveBayes, C4.5 decision tree, and SVM (SMO) algorithms to explore the performance of every algorithm on each feature set. The result is shown in Table 5.3:

	Features	Number of	Pres., Freq. Or Other	BayesNet	Naïve Bayes	C4.5	SVM

		Features	Weighting Method				
1	unigrams	1447	Pres.	**57.9962**	48.5761	47.653	54.4564
2		1447	Freq.	**57.9962**	48.6094	48.641	56.7004
3		1447	tf/idf	**57.9962**	49.0809	48.7042	56.9532
4	weighted unigrams	1447	Frequency*SO(w)	**57.8594**	49.5777	49.6726	55.5196
5		1501	Presence_POS	**57.5221**	50.7067	49.2536	55.4325
6		1501	Frequency_POS	**57.5221**	49.2701	48.6521	56.1215
Avg				57.8154	49.3035	48.7628	55.8640
7	Feature set 1 and its variations	1447	Pres.	**56.2234**	47.2530	45.3902	54.7789
8		1447	Freq.	**56.2234**	47.2047	43.2245	55.0530
9		1447	tf/idf	**56.2234**	47.8832	44.3302	55.2306
10		1447	Frequency*SO(w)	**56.4530**	47.0304	45.061	56.1146
Avg				56.2808	47.3428	44.5015	55.2943
11	Feature set 2 and its variations	1447	Pres.	**58.1103**	49.2057	45.081	57.2305
12		1447	Freq.	**58.1103**	49.3109	46.6321	57.9671
13		1447	tf/idf	**58.1103**	49.4605	46.1232	58.0015
14		1447	Frequency*SO(w)	58.1209	49.8021	46.5433	**58.4503**
Avg				58.1130	49.4448	46.0949	57.9124
15	Feature set 3 and its variations	1451	Pres.	**58.121**	49.2579	46.0947	57.2809
16		1451	Freq.	**58.121**	49.3020	47.3509	57.9921
17		1451	tf/idf	**58.121**	49.7721	46.5617	57.9734
18		1451	Frequency*SO(w)	58.3215	49.4559	46.7539	**58.4203**
Avg				58.1711	49.4470	46.6903	57.9167

Table 5.3 10-fold cross-validation learning accuracies, in percent, on 18 Feature Sets for four learning algorithms; the boldface in each line is the best performance of four algorithms (Avg is the average value of the accuracies in each section)

Table 5.3 shows thatoverall, BayesNet classifier and SMO outperformed the other two algorithms. Among all the 18 datasets, 16 best results are produced by BayesNet and only two best results are contributed by SMO algorithm. The machine learning results obviously surpass the random-choice baseline around 20%. (We did not repeat the random scoring in this section, and take the result of section 5.2.4 as a reference. In that section, we obtained 20.6384% and 20.8281% accuracy using random scoring with the five star scoring system).

Furthermore, the classification result by BayesNet and SMO outperforms the unsupervised learning method with which we achieve 52.5215% and 50.7269% accuracy based on the Feature Set 1 and feature Set 2. On Feature Set 2 and Feature Set 3, the Naïve Bayes classifier

also presents comparative accuracy with our unsupervised learning method, so we keep it and explore it further in next section 5.3.2 when trying to improve the learning performance. On the other hand, J48 (C4.5) decision tree get the lowest accuracy of prediction on all the 18 datasets, its accuracy is about, on average, five percent worse than the second lowest accuracy by Naïve Bayes. Consequently, we do not continue the exploration on C4.5 decision tree classifier in coming sections of our experiments.

5.3.1.2 Analysis: Results by Algorithm

The performance of C4.5 decision tree The reason why C4.5 only gets the lowest accuracy and outperformed by the simple algorithms such as NaiveBayes and BayesNet relates to its characteristics:

A. In our experiments, all the attributes are numeric (real) type, and they represent the appearance, frequency, tf/idf value, SO_WN score and so on. However, decision trees are inherently more suited to using nominal attributes than numeric attributes. When splitting branches on attributes, the nominal attribute can only be tested once on the path from root to the leaf, while numeric attributes might be tested more than once. This results in the tree becoming too complex and difficult to understand, and the scattered attributes which are along the path rather than located together are actually a kind of overfitting phenomenon. Although decision trees can be adjusted to execute multi-way tests for numeric attributes, it is unfeasible for our datasets because of the large number of attributes and the wide range of attribute values. Of course, prediscretizing the attribute into nominal type seems to be an effective solution, but it improves the other three classifiers and not C4.5; we will explain the problem caused by prediscretization for C4.5 in next section.

B. Without pruning, the complex decision trees are often outperformed by simple decision trees due to the generation process of subtrees. The influential decision tree C4.5 uses postpruning during its tree-building process. In the postpruning, there are two different strategies: subtree replacement and subtree raising.

Unfortunately, in our scenario, both operations negatively affect the performance of C4.5. Subtree replacement selects some subtrees and replaces them by single leaves. The idea behind subtree replacement is sacrificing the local accuracy on the training set (because it is hard to make all leaf nodes pure) to increase the accuracy on the independent test set. Actually, its essential methodology is using strict pruning to counteract the effect of possible overfitting resulting from the model building process. However, this operation probably prevents C4.5 decision tree from completely developing on 1447 features, and then prunes the growing subtrees too early.

On the other hand, the other operation of *subtree raising* is also likely to harm the performance of C4.5 in our experiments. Because this strategy intends to raise the subtree of the most popular branch which has more training examples, it is apt to benefit the majority class. Especially in our imbalanced dataset, the subtree which includes more majority class nodes of score 1 is easy to be over weighted and raised. In contrary, the minority class becomes sparser than their actual proportion in original dataset. In other words, the classification result actually is skewed further.

To sum up, decision tree is not a feasible classifier for our classification task based on numeric features, and so we do not explore it further.

Naïve Bayes classifiers vs. C4.5 BayesNet produces the best prediction among four classifiers, and even the basic Naïve Bayes algorithm also surpasses C4.5 decision tree although its performance is not comparative with BayesNet (We are discussing Naïve Bayes in this section and BayesNet in next section).

Similar to logistic regression model, the Naïve Bayes classifier produces probability estimates rather than only predictions. For each class label, it estimates the probability that a given instance belongs to that class, and this feature makes the simple Naïve Bayes algorithm more reasonable and feasible for our classification. The scoring of reviews, sometimes, is more like a regression task than a simple five class classification job. Actually, sentiment of customer reviews often is essentially a continous spectrum instead of five discrete integer scores as

shown in Figure 1.1. For example, the concept of "half stars" in movie reviews is used on many movie review websites because reviewers feel the choice between a mere five ratings to be too rough and strict. Even on some movie review websites, the reviewers can accurately specify the number for the first decimal place and rate a movie by decimal fractions such as '4.5', '3.7' and so on. Because of the gradually increasing precision of scoring, the probability estimation becomes a more and more promising approach for predicting numeric classes.

In addition, Naïve Bayes estimates the conditional probability distribution of class values given the values of all features. This classification model represents the conditional distribution more concisely and comprehensibly than decision trees, because the decision tree uses a very simplistic way, in which the probabilities are computed by the relative frequency of each class in a leaf and generate a decision list by examining the samples that a particular rule covers.

Furthermore, all values of the attributes in our experiments are real numbers. Decision tree learners, on the other hand, deal with numeric attributes on a local basis, examining attributes at each node of the tree when it is being constructed to see whether they are worth branching on—and only at that point deciding on the best place to split continuous attributes. Consequently, the decision trees tend to fragment the training set into smaller and smaller pieces, which produce less reliable probability estimations. Meanwhile, this algorithm results in many replicated subtrees.

As a result, not only do the training and testing processes become time-consuming, but also the learning performance is worse than the result by Naïve Bayes. The results in Table 5.3 displayed that averagely the accuracy of Naïve Bayes surpasses the accuracy of C4.5 by 2.37 percent.

BayesNet BayesNet performs very well on all datasets. It contributes all the highest accuracies of prediction except on two datasets which are shown in line 14 and line 18 of Table 5.3.

Like the Naïve Bayes algorithm, BayesNet is based on the assumption of conditional

independence, and both of them are alternative ways of representing a conditional probability distribution of instances. However, Naïve Bayes can only represent simple distributions, so its accuracy declines when dealing with multiclass problem on large set of attributes. On the other hand, BayesNet constructs a network in which every attribute has a node, and organizes all nodes into a chain with respect to probability distribution relationships among all attributes. Additionally, this chain represents a causal effect by which BayesNet multiplies all probabilities on the chain together and obtains the final probability. Our feature vectors of reviews, especially on the Feature Set 2 and Feature Set 3, take advantage of this structure: because both of these feature sets involve WordNet selected features, they agree with the attribute-independent assumption much better than Feature Set 1 and unigram feature set do. Consequently, the accuracy on both Feature sets 2 and 3 are 2% higher than on the unigram feature set and Feature Set 1. Furthermore, because BayesNet well utilizes the causal information that other classifiers cannot use, it surpasses other classifiers on sixteen feature sets of a total of eighteen datasets as shown in Table 5.3.

When running BayesNet algorithm of WEKA, we use K2 as our default search algorithm, because some other algorithms, such genericSearch, are too time-consuming; and the rest, like TAN method, are not optimal. In our 10-fold cross-validation run, the result using the TAN search approach is far worse than using the K2 approach, so we do not present this trial result here. Furthermore, we chose the default values for all other configuration parameters, i.e. set the options of 'maxNrOfParents' to 1, and left the 'markovBlanketClassifier' and 'randomOrder function' unchecked.

SMO SMO presented the second best performance of all classifiers, but only achieved the best accuracies on line 14 and line 18 of Table 5.3. On the Feature Set 2 and 3, SMO is comparable with BayesNet, but on Feature Set 1 and unigrams, it is outperformed by BayesNet.

At the beginning we expected SMO to contribute the best results on most of the datasets, because we thought that support vector machine is extraordinarily appropriate to our scenario because we need an approach to deal with the numeric attributes using a method similar to linear regression but without linear class boundaries, and it happens that support vector

machine satisfies these requirements. However, why does it not outperform BayesNet?

The first reason is that for our multi-class classification problem, SMO adopts pairwise classification by Hastie and Tibshitani [63]. This is a 1-vs-1 method which builds a logistic model using the pairwise coupling approach. Actually, this method can not completely avoid the effect of imbalanced datasets which we will discuss in detail in section 5.3.3.2. The skewed datasets cause the 1-vs-1 method to be biased towards the majority class. To obtain the proper probability estimates, Hastie et al. use the option that fits logistic regression models to the output of support SVM, while the predominance of the instances of class 1 still make the SMO algorithm biased towards the class 1, as shown in Figure 5.7:

```
=== Detailed Accuracy By Class ===

TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
  0.786     0.28       0.7       0.786     0.741      0.821    1.0
  0.163     0.106      0.2       0.163     0.18       0.578    2.0
  0.198     0.064      0.223     0.198     0.209      0.618    3.0
  0.306     0.085      0.347     0.306     0.326      0.753    4.0
  0.564     0.097      0.58      0.564     0.572      0.846    5.0

=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 1129  177   66   28   36 |    a = 1.0
  272   72   29   38   31 |    b = 2.0
  100   43   53   40   32 |    c = 3.0
   51   35   47  125  150 |    d = 4.0
   61   33   43  129  344 |    e = 5.0
```

Fig 5.2 The learning result of SMO based on imbalanced data

As presented in Table 5.2, the precision on Class 1 (the largest class) and class 3 (the smallest class) are 70% and 22.3% respectively. Although the difference of accuracy between them is not as big as that of the result of Naïve Bayes algorithm, it is still remarkable.

The second reason is that in our experiments, we chose the polynomial kernel which uses the

dot product between vectors to classify instances, as the kernel of SMO. When choosing the polynomial kernel, the upper bound of the coefficients α_i is very important, but unfortunately it is fixed in WEKA, so we could not change its value.

On the other hand, the alternative choice *radial basis function kernel* should be more appropriate to our datasets, because it is simply a type of neural network and implements a multilayer perceptron with no hidden layes. Actually, this is another type of neural network, and it can substitute for logistic regression and produce much better performance. Unfortunately, this kernel requires a large amount of memory, and we were unable to use it on our data set as it needed more than 2GB of memory for WEKA. However, it would be interesting to experiment with this approach in the future.

Originally, because BayesNet, Naïve Bayes, C4.5 and SMO have been reported achieve accuracies around 90% in topic classification and 80~83% in binary "Thumbs Up" or "Thumbs Down" classifications, we experimented with them in this section and expected good performance from them. However, the results of our experiments suggested: first, the sentiment classification is more difficult than the text categorization task, and second, the multi-class classification is more complex than two class classification problem.

As a result, due to the good performance by BayesNet and SMO, we continue the improvement experiments in coming sections. We also keep Naïve Bayes algorithm because on some datasets its performance is competitive, so we want to get a view whether the improvement experiments are effective for it. On the other hand, the C4.5 algorithm is abandaned due to having the lowest accuracy among the four classifiers.

5.3.1.3 Analysis: Results by Feature Set

Unigrams vs. Feature Set 1 The classification accuracies resulting from using only unigrams as features are shown from line 1 to line 6 of Table 5.3 and the results using Feature Set 1 are from line 7 to line 10. As a whole, by all four classifiers BayesNet, Naïve Bayes, C4.5 and

SMO, results on unigrams clearly surpass the results on Feature Set 1.

All the best results are generated by BayesNet algorithm, and the results on unigrams beat all results on Feature Set 1. On each weighting method of unigrams and feature set 1, the results based on word presence, frequency and *tf/idf* have no difference between each other. In the six weighting method s of unigrams, the accuracy 57.9962% of the first three weighting methods (line 1~3) beat the results: 57.8594%, 57.5221% and 57.5221% of the other three weighting methods (line 4~6) of Frequency**SO*(*w*), Presence_POS and Frequency_POS.

Lets recall the definitions of Frequency**SO*(*w*), Presence_POS and Frequency_POS. They are different weighting methods used for the feature selection process: Frequency**SO*(*w*) is the product of the unigram frequency and the sentiment orientation score described by formula (4.1) in section 4.3; Presence_POS is the count of unigrams using appended POS tags of every term via GPoSTTL lemmatizer, and then mapped the POS tags onto nouns, verbs, adjectives and adverbs using the Appendix B. Similar to Presence_POS, Frequency_POS also append POS tags when counting unigrams, but values of Frequency_POS features denote the frequency instead of the presence.

Unfortunately, using weighting methods does not improve the learning result. On contrary to our expectations, compared to unigrams without weighting information (line1~line3), the weighted unigrams (line 4~line 6) do not contribute higher accuracies on any of the four classifiers. Only the Presence_POS on line 5 of Table 5.3 obtained slight increase over all unigrams without weighting using BayesNet (57.5221%) and C4.5 (49.2536%) vs., and the performance of other combinations were almost unchanged. For BayesNet, which has the highest accuracies of the four classifiers, we found the result 57.8594% based on Frequency*SO(w) is comparable to that of unigrams withough weighting, 57.9962%, so we continue to use it as one candidate feature set.

Therefore, in the following experiments, we only keep four weighting options 'Pres.', 'Freq.', 'tf/idf' and 'Frequency**SO*(*w*)' for Feature Set 1~ Feature Set 3.

Obviously, including the POS information does not help the classifiers. First, there are not many sentiment words ranked top in the high frequency sentiment word lists by different POS (in our experiments only 54 words). In adition, the sentiment orientation does not change between their different POS, and the degrees of the subjectivity of their different POS are almost same. Therefore, distinguishing POS for few sentiment words does not help improve the learning result, and it actually hurts a bit the performance.

Feature Set 1 has 1447 features including 792 unigrams and 665 bigrams. Intuitively, we expected that the performance on feature set 1 to be better than on unigrams, because the modified adjectives and adverbs are included and the refined sentiment bigrams should more accurately indicate the sentiment orientation than mere unigrams. Nevertheless, when comparing the results from line 7 to line 10 with results from line 1 to line 6, as shown in Table 5.3, we find that the results using Feature Set 1 with bigrams are worse than the results using unigrams alone

This result implies that bigrams do not necessarily benefit the learning of sentiment analysis. The bigrams of Feature Set 1 is simply captured by GI and SO-PMI algorithms, the accuracies from line 7 to line 10 show that these 665 bigrams yield less useful information than the 665 unigrams they substitute for in the unigrams feature sets.

In Table 5.3, on Feature Set 1, the results classified by Naïve Bayes, C4.5 and SMO present the same trend consistent with BayesNet: all the accuracies are lower than that based on unigrams; especially the result of C4.5 declined 4.26%, which is more remarkable than the 1.96% decline of Naïve Bayes and the 0.57% decline of SMO. The reason for the difference between them is that decision trees are more sensitive to the information brought by special features (attributes) when these features are used as high level nodes in a decision tree. In addition, the non-linear model of the SMO and the Bayes's rule of Naïve Bayes are less impacted by the relatively informationless bigrams, so they are not so sensitive to the difference resulting from bigrams as the decision tree is.

To conclude: First, consistent with the observation of Bo Pang et al [10] and P.D.Turney [14],

the bigrams generated by GI dictionary and SO-PMI algorithm do not benefit classifiers more than unigrams do; Secondly, C4.5 is more apt to be affected by the quality of features than Bayes Net, Naïve Bayes, and SMO .

Feature Set 2 and Feature Set 3 Unlike the results of Feature Set 1, the performance of Feature Set 2 and Feature Set 3 surpass that of unigrams.

Recall that both Feature Set 2 and 3 include 792 unigrams and 655 bigrams. In order to investigate whether using WordNet information could provide great improvement for the four classifiers, we use SO_WN score to extract 396 of them rather than using the SO-PMI method to select all 792 unigrams. The difference between Feature Set 2 and Feature Set 3 is that Feature Set 3 includes four informative synthetic features Average_PMI, Average_lesk, Average_hso and Average_jcn.

As can be seen from line 11 to line 18 of Table 5.3, better performance is achieved by BayesNet, Naïve Bayes and SVM algorithms on Feature Set 2 and 3:

The accuracy of BayesNet on Feature Set 2 is 0.30% and 1.83% higher than the accuracy on the unigram feature set and on Feature Set 1; the accuracy of BayesNet on Feature Set 3 is 0.36% and 1.89% higher than the accuracy on the unigram feature set and on Feature Set 1.

Overall on Feature Set 2, the accuracy of Naïve Bayes is 0.14% and 2.10% higher than on the unigram feature set and on Feature Set 1; On Feature Set 3, the accuracy of Naïve Bayes is 0.14% and 2.10% higher than on the unigram feature set and on Feature Set 1.

on Feature Set 2, the accuracy of SMO is 2.048% and 2.618% higher than on the unigram feature set and on Feature Set 1; On Feature Set 3, the accuracy of SMO is 2.052% and 2.622% higher than on the unigram feature set and on Feature Set 1.

To sum up, we can conclude that all classifiers did better on Feature Sets 2 and 3 than on Feature Set 1 or unigrams.

Based on BayesNet and C4.5, we speculate that this result may indicate:

First, the performance on feature set 2 seems to beat that on the unigram feature set, which means the combination of SO_WN and SO-PMI feature selection methods is a little better than using unigrams only--but this conclusion remains to be verified;

Secondly, compared to Feature Set 1, the learning performance of BayesNet and SMO on Feature Set 2 and 3 is improved obviously. This result implies that the involvement of SO_WN benefits BayesNet and SMO classifiers;

Last but not least, the accuracy on feature set 3 shows slight improvement over Feature Set 2, which is supposed to have much improvement by the four additional compositive features Average_PMI, Average_lesk, Average_hso and Average_jcn, but it does not as expected.

On the other hand, the C4.5 decision tree presents different results:

Averagely, on Feature Set 2, the accuracy of C4.5 is 2.67% lower and 1.59% higher than on unigrams and on Feature Set 1 respectively; On Feature Set 3, the accuracy of C4.5 is 2.07% lower and 2.19% higher than on unigrams and on Feature Set 1 separately. This result means that C4.5 on Feature Set 2 outperforms on Feature Set 1, but C4.5 on Feature Set 2 is surpassed by C4.5 on unigrams.

Based on C4.5, we find that this result may indicate:

First, unlike the results of BayesNet, Naïve Bayes and SMO, by C4.5, the accuracy of Feature Set 2 declined compared to that of the unigram feature set; C4.5 is very abnormal because its accuracy on Feature Set 3 is 2.07% lower than that on unigrams. Due to the exception by C4.5, so far we cannot confirm whether incorporating bigrams with unigrams outperforms using unigrams feature set alone. We will trace this comparison in

the following experiments.

Secondly, compared to Feature Set 1, the learning performance of Naïve Bayes and C4.5 on Feature Set 2 and Feature Set 3 is improved greatly. This result is consistent with BayesNet and SMO, and proved that the involvement of SO_WN surely benefits these three classifiers;

Finally, same as the above results by BayesNet and SMO, the accuracy of Naïve Bayes and C4.5 on feature set 3 is slightly better than the accuracy on Feature Set 2. Therefore, we can infer that the synthetic features of Average_PMI, Average_lesk, Average_hso and Average_jcn surely improve the classifiers.

In a word, based on the previous study, from Table 5.3, we can conclude:

A. Using bigrams to replace a same amount of unigrams does not necessarily improve the learning results of the four classifiers we used. Obviously, from line 1 to line 10 of Table 5.3, we can find that the better performance is obtained by unigrams alone, not by mixed feature set of unigrams and bigrams. This is consistent with the observation of Bo Pang et al [10] with respect to the binary sentiment classification, in which they conclude that bigrams are not effective at capturing contextual information although the context is very important.

   On the other hand, the result through line 1 to line 10 is in direct opposition to the study of Alistair Kennedy and Diana Inkpen [11], in which the negation words, intensifiers and diminishers played important role for improving the performance of movie reviews classification. This difference perhaps results from the possibility that the simple bigram extraction method we used, which relies on P.D.Turney's [14] study, is too coarse to accurate capture all the sentiment bigrams using valence shifters.

B. The features selected using WordNet are better than the features selected using SO-PMI. Therefore, the classification benefits more from the features which are selected using

WordNet information than the selected features using SO-PMI algorithm only.

C. The additional synthetic features of Average_PMI, Average_lesk, Average_hso and Average_jcn slightly benefit classification, although the difference is small.

D. C4.5 presents an inconsistent trend compared to the other three classifiers: not only is its performance the worst of the four classifiers, but also its effect on our investigation (using bigrams, extracting unigrams with WordNet, and using synthetic features) is opposite to the common pattern. We think that the particularity of the structure of decision tree accounts for its abnormal result, and C4.5 is not a good algorithm for the task of multiclass classification for customer reviews. Therefore, to avoid the disturbance resulting from C4.5, we give it up in following sections about improvement experiments.

5.3.2 Improvements

According to the discussion in section 5.3.1, the C4.5 is a relatively poor classifier for our sentiment classification task, so we do not use it in our exploration of improvement experiments. Furthermore, because the weighting methods of Presence_POS and Frequency_POS (line 5~6 in Table 5.3) are not especially beneficial to the classification, we also delete them from our candidate feature sets.

In addition, the result of Table 5.3 strongly implies that combining the SO-PMI and SO_WN methods to extract unigrams is very effective for increasing the accuracy of classification; on the other hand, because adding bigrams does not improve the performance very much on Feature set 2 and 3, and Feature Set 1 even worse than unigrams set, intuitively we speculate that using SO-PMI and SO_WN together to select unigrams instead of relying on GI dictionary may yield better results. Therefore, we exchange the unigrams dataset with new unigram features extracted by SO-PMI and SO_WN together. However, we keep the size of a total of 1447 unigrams unchanged. In the 1447 unigrams, 723 are SO-PMI selected, and the other 724 are SO_WN selected.

5.3.2.1 Meta Learning: Making use of ordinal information in five star scoring

As discussed in section 4.5.2, we incorporate the ordinal information of class attributes into the standard classification learner, to make use of the ordering nature of five star grading schemes. The goal of this improvement is to check whether meta-learning methods can take advantage of the ordinal information hiding in the scores of bank reviews.

Table 5.4 shows the learning result using meta-learning of ordinal classifier based on standard BayesNet, NaiveBayes and SVM classifiers.

	Features	Number of Features	Pres.,Freq. Or Other Weighting Methods	Standard Base Algorithm used By Meta Classifier		
				BayesNet	Naïve Bayes	SVM
1	Unigrams (SO-PMI+ SO_WN)	1447	Pres.	**60.292**	51.0936	52.1808
2		1447	Freq.	**60.292**	51.1211	52.3004
3		1447	tf/idf	**60.292**	51.6055	52.9532
4		1447	Frequency_POS	**59.8030**	51.4002	51.8355
Avg				60.1698	51.3051	52.3178
5	Feature set 1 (and its variations)	1447	Pres.	**57.6673**	49.6531	50.7789
6		1447	Freq.	**57.6673**	49.2049	50.0530
7		1447	tf/idf	**57.6673**	50.1856	50.2306
8		1447	Frequency*SO(w)	**57.8021**	49.537	50.1146
Avg				57.701	49.6452	50.2948
9	Feature set 2 (and its variations)	1447	Pres.	**61.9942**	50.9057	51.2305
10		1447	Freq.	**61.9942**	50.8109	51.9671
11		1447	tf/idf	**61.9942**	51.216	52.0015
12		1447	Frequency*SO(w)	**62.1209**	50.8021	51.4503
Avg				62.0259	50.9337	51.6624
13	Feature set 3 (and its variations)	1451	Pres.	**62.3233**	51.1579	51.2809
14		1451	Freq.	**62.3233**	51.3022	51.9921
15		1451	tf/idf	**62.3233**	51.7721	51.9734
16		1451	Frequency*SO(w)	**62.3215**	50.9832	51.4203
Avg				62.3229	51.3039	51.6667

Table 5.4 The learning results using meta-learning of ordinal classifier

The ordinal multiclass classification produced the best result based on BayesNet classifier. Compared to Table 5.3, on unigram feature set, Feature Set 1, 2, and 3, this meta classifier obtained significant improvement. For comparison, from Table 5.3 and Table 5.4 we refine the average accuracy by three algorithms on four feature sets into Table 5.5.

	Feature Set	Standard Classifier			Meta Learning Based on Standard Classifier		
		BayesNet	Naïve Bayes	SVM	BayesNet	Naïve Bayes	SVM
1	Unigrams	**57.8154**	49.3035	55.864	**60.1698**	51.3051	52.3178
2	Feature set 1	**56.2808**	47.3428	55.2943	**57.701**	49.6452	50.2948
3	Feature set 2	**58.113**	49.4448	57.9124	**62.0259**	50.9337	51.6624
4	Feature set 3	**58.1711**	49.447	57.9167	**62.3229**	51.3039	51.6667
Average		**57.5951**	48.8845	56.7469	**60.5549**	50.7970	51.4854

Table 5.5 The average accuracy by three algorithms on four feature sets

Table 5.5 shows that, as we expected, overall meta-learning results surpass the results of the standard classifiers. With both learning machanisms, BayesNet achieves the best performance on all feature sets. Especially on unigrams, the line 1 in Table 5.5, meta-classifier based on BayesNet obtains 2.35% increase of accuracy compared to standard BayesNet; On the other hand, the average accuracy of four feature sets get a 2.96% improvement by the meta classfier.

In our experiments, we adopted the *paired t-test* to evaluate the statistical significance. For the meta-learner using ordinal information, the t-test results of WEKA are displayed as follows:

```
Dataset                  (1) bayes.Na | (2) meta.
-----------------------------------------------
sentiment_analysis       (200)   58.00 |   60.29 v
sentiment_analysis       (200)   58.00 |   60.29 v
sentiment_analysis       (200)   58.00 |   60.29 v
sentiment_analysis       (200)   57.86 |   59.80 v
sentiment_analysis       (200)   56.22 |   57.67 v
sentiment_analysis       (200)   56.22 |   57.67 v
sentiment_analysis       (200)   56.22 |   57.67 v
sentiment_analysis       (200)   56.45 |   57.80
sentiment_analysis       (200)   58.11 |   61.99 v
sentiment_analysis       (200)   58.11 |   61.99 v
sentiment_analysis       (200)   58.11 |   61.99 v
```

```
sentiment_analysis        (200)    58.12 |    62.12 v
sentiment_analysis        (200)    58.12 |    62.32 v
sentiment_analysis        (200)    58.12 |    62.32 v
sentiment_analysis        (200)    58.12 |    62.32 v
sentiment_analysis        (200)    58.32 |    62.32 v
-------------------------------------------------
                                (v/ /*) |   (15/1/0)
```

Except the learning result on the Unigrams of Frequency*SO(w), on all other 15 Feature Sets, the performance of ordinal meta-classifier is statistically better (marked with 'v') than the baseline scheme (in our case, standard Naïve Bayes) at the significance level specified (0.05 in our experiment).

Hereafter, and throughout the following sections, when we refer to something as "significant", we always mean statistically so with respect to the paired t-test, $P < 0.05$. Furthermore, the "comparison field" parameter of WEKA is "Percent_correct".

Therefore, we can conclude that, as described above, and coincident with what we expected, overall Naïve Bayes meta-learning results surpass the results of the standard Naïve Bayes classifiers.

Similar to the previous result in Table 5.4, the performance of Naïve Bayes is not as good as that of BayesNet. BayesNet and Naïve Bayes both adopt the Bayes rule to calculate the probabilities from the value of attributes, and both of them to some extent violate the prerequisite conditional-independence assumptions because neither unigrams nor bigrams are necessarily independent; In opposition to this assumption, depending on the distance between them and the syntactical structure of sentences, more or less, potential relationships almost always exist among unigrams, bigrams or even between some unigrams and bigrams. In fact, unigrams and bigrams are very often dependent.

However, as we mentioned before in section 5.3.1.2, when dealing with classification in a specific domain, BayesNet can represent causal effects by edges and benefits from the network structure. Therefore, BayesNet is less affected by the weaknesses of the factual dependence

between attributes than Naïve Bayes.

Because of its relatively poor performance, we do not further explore Naïve Bayes.

On the other hand, one exceptional phenomenon happened when using the ordinal meta-classification based on SMO algorithm: the performance of meta-classifier using SMO declined dramatically from the average 56.7469% to an average 51.4854%, as shown in line 5 of Table 5.5. This deterioration of performance of meta-learning results from the scenarios shown in Figure 5.3 and 5.4:

```
=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
    0    0   12  193   69 |    a = 3.0
    0    0   32  360   24 |    b = 4.0
    0    0    3  243  189 |    c = 2.0
    0    0   52  567   17 |    d = 5.0
    0    0   13  309 1081 |    e = 1.0
```

Fig 5.3 Confusion Matrix of Meta-Leaning by SMO

(on Feature Set 2 *tf/idf* weighting method in line 11 of Table 5.4)

```
=== Confusion Matrix ===

   a   b   c   d   e   <-- classified as
   0   7 133 122  48 |   a = 3.0
   0  15 166 228  15 |   b = 4.0
   0   3  83 175 161 |   c = 2.0
   0  23 151 535  13 |   d = 5.0
   0   3 123 312 848 |   e = 1.0
```

Fig 5.4 Confusion Matrix of Meta-Leaning by SMO

(on Feature Set 1 frequencySO*(*w*) weighting method in line 8 of Table 5.4)**

In our sentiment classification task, there are five classes. According to the description in section 4.5.2, the ordinal meta-classifier splits the multiclass classification into four binary SMO classifiers. In the process of building each binary SMO model, SMO uses Linear Kernel, which is chosen as default by WEKA, trying to find the non-linear boundaries used to classify samples. At this time, because of the imbalanced data distribution described in Table 4.6, the training process of SMO dramatically biases to the majority classes, and assign very unreasonable weights (approximate 0) for all attributes. These abnormally small weight values, which are shown in Figure 5.5, cause the meta-classifier to fail to classify any samples into the minority classes.

```
Classifier for classes: neg_2-last, pos_2-last

BinarySMO

Machine linear: showing attribute weights, not support vectors.

          0       * (normalized) avoid
 +       -0.0001  * (normalized) bad
 +       -0.0001  * (normalized) best
 +        0.0001  * (normalized) beware
 +        0.001   * (normalized) bonus
```

Fig 5.5 BinarySMO Modeling Result for Minority Class

(on Feature Set 1 frequency weighting method in line 8 of Table 5.4)

Take the modeling process corresponding to the Figure of 5.3 (on Feature Set 2 *tf/idf* weighting method in line 11 of Table 5.4) as example, when building the meta-classifier based on SMO, about the Class 3, shown as the first column in Figure 5.3 (corresponding to line 3 in Table 4.6, a total of 268 samples) and Class 4, shown as the second column in Figure 5.3 (corresponding to line 4 in Table 4.6, a total of 408 samples), we find that the number of samples of both Class3 and Class4 are 0. Obviously, this result is strongly biased towards the majority and totally ignores the minority.

This misclassification directly caused the performance of meta-classifier to drop down from the average 56.7469% to an average 51.4854%, as shown in Table 5.5.

We will try to use re-sampling to solve the imbalanced classification problem; however, it does not help the meta-learning based on standard SMO. Furthermore, other kernel algorithms provided by SMO, such as *puk*, *RBF* kernel etc., are all extraordinary time-consuming and resource-consuming. The learning process relying on other SMO kernels either exhausted tens of hours (even longer when using re-sampling and discretization) or caused the overflow of memory although we have already allocated 2 GB memory to WEKA.

Therefore, we do not use SMO any longer as our candidate classifier to explore other approaches for improvement, but only use BayesNet in the following experiments. In fact, for discovering the effectiveness of Re-sampling, discritization etc., BayesNet is convincing enough.

5.3.2.2 Re-sampling

As discussed in section 4.5.3.1, when there are many more instances of some classes than others in a dataset, we consider this dataset to be imbalanced. Generally, when learning from datasets with imbalanced class distributions, machine learning algorithms tend to produce biased classifiers because standard classifiers tend to be overwhelmed by the large classes and ignore the small ones. In this section, we experiment with the re-sampling method to alleviate the problems resulting from skewed data.

	Features	Number of Features	Pres.,Freq. Or Other Weighting Methods	Ordinal meta-classifier based on BayesNet
1	unigrams	1447	Pres.	**60.2086**
2		1447	Freq.	**61.7358**
3		1447	tf/idf	**60.9962**
4		1447	Frequency_POS	**60.0103**
Avg				60.7377
5	Feature set 1	1447	Pres.	**58.2577**
6		1447	Freq.	**59.1335**

7	(and its variations)	1447	tf/idf	**59.4346**
8		1447	Frequency*SO(w)	**58.1933**
Avg				58.7548
9	Feature set 2 (and its variations)	1447	Pres.	**62.9961**
10		1447	Freq.	**63.1244**
11		1447	tf/idf	**63.2332**
12		1447	Frequency*SO(w)	**63.5121**
Avg				63.2165
13	Feature set 3 (and its variations)	1451	Pres.	**63.5675**
14		1451	Freq.	**63.9306**
15		1451	tf/idf	**63.8430**
16		1451	Frequency*SO(w)	**63.3524**
Avg				63.6734

Table 5.6 Learning Result of BayesNet Using Re-sampling

As shown in Table 5.6, the learning results of 'Pres.', 'Freq.', and 'tf/idf' (for example, the line 1 ~ line 3) do not stay the same any longer on each feature set. Because the re-sampling does not always collect the same redundant samples when building models, so it is natual that the accuracies of different weighting strategies are distinguished from each other.

	Feature Set	Meta Learning without Re-sampling Based on standard BayesNet	Meta Learning with Re-sampling Based on standard BayesNet
1	Unigrams	60.1698	**60.7377**
2	Feature set 1	57.701	**58.7548**
3	Feature set 2	62.0259	**63.2165**
4	Feature set 3	62.3229	**63.6734**
Average		60.5549	61.5956

Table 5.7 Average Learning Accuracies of BayesNet Using Re-sampling

Table 5.7 shows that, overall, the meta-learning with re-sampling outperformed that without re-sampling. The data in Table 5.7 are extracted from lines of average accuracies of each feature set in Table 5.4 and Table 5.6. As a whole, about the average accuracy on all 16 feature sets, re-sampling bring a 1.04% percentage increase to the ordinal meta-learning.

=== Confusion Matrix ===							=== Confusion Matrix ===						
a	b	c	d	e	<-- classified as		a	b	c	d	e	<-- classified as	
1283	31	38	31	53	|	a = 1.0	**1363**	46	12	42	33	|	a = 1.0
325	**18**	25	23	51	|	b = 2.0	282	**51**	15	24	37	|	b = 2.0
123	12	**39**	38	56	|	c = 3.0	111	13	**34**	35	53	|	c = 3.0
91	21	29	**79**	188	|	d = 4.0	74	16	23	**90**	209	|	d = 4.0
90	12	17	87	**404**	|	e = 5.0	104	15	5	28	**449**	|	e = 5.0

Ordinal meta-learning without Re-sampling

Ordinal meta-learning with Re-sampling

Fig 5.6 The comparison of instance distribution of meta-learning between classifiers with and without re-sampling.

(Based on 1447 unigrams of word presence)

Using confusion matrices, Figure 5.6 shows the distribution of samples about the meta-learning with and without re-sampling. Both matrices are based on the 1447 unigrams weighted by word presence: the right matrix implemented re-sampling, but the left one did not. As can be seen from Figure 5.6, with re-sampling, not only do the numbers of True Positive instances increased, but also the distribution of them became more balanced than that of without re-sampling (the classifier was not as biased towards the majority classes as it was without resampling).

	Score	No. of Insts.	Meta-learning without re-sampling			Meta-learning with re-sampling		
			TP Rate	FP Rate	Precision	TP Rate	FP Rate	Precision
Class1	1.0	1436	0.893	0.364	0.671	0.911	0.342	0.705
Class2	2.0	442	0.041	0.028	**0.191**	0.125	0.033	**0.362**
Class3	3.0	268	0.146	0.038	**0.264**	0.138	0.019	**0.382**
Class4	4.0	408	0.194	0.065	**0.306**	0.218	0.047	**0.411**
Class5	5.0	610	0.662	0.136	0.537	0.747	0.13	0.575

Table 5.8 The comparison of detailed accuracies of meta-learning between classifiers with and without re-sampling.

(Based on 1447 unigrams of word presence)

In addition, the Table 5.8 shows the comparison of detailed accuracies between the two results. It is worth noting that the precision of class 2, class 3 and class 4 (line 2, 3 and 4), which are minority classes, obtained obvious improvement after implementing re-sampling (marked by boldface); Meanwhile, the precision of majority classes did not decrese.

Finally, it is inevitable that re-sampling usually is, to some extent, related to overfitting. To minimize the overfitting, we did not force the re-sampling bias towards a uniform class; In other words, when using WEKA to implement our experiments, we did not use the 'InvertSelection' and 'Replacement' options, and we set the 'randomSeed' value to its lowest allowed value '1' to avoid the excessive overfitting (generally, the higher the velue of the parameter 'randomSeed', the more serious the overfitting). Because all re-sampling options are setup to reduce the overfitting, the above result surely indicates that re-sampling is a useful and effective approach to eliminate or at least diminish the effect resulting from skewed distribution of instances.

5.3.2.3 Discretization

In general, discretization of numeric attributes is absolutely essential when the task involves numeric attributes in modeling process whereas the chosen learning method can only handle categorical ones. In our experiments, the attributes are numeric (real) values, but all the three algorithms BayesNet, Naïve Bayes and SMO can deal with numeric features, so it seems that discretization is not necessary. In fact, however, these three learning methods not only produce better results but also work faster when the datasets are prediscretized. Especially, a single run of SMO algorithm using re-sampling based on the 1447 unigrams lasted 89 hours when we adopted 10-fold cross-validation, while this runtime is shortened to 22 hours after the discretization is used.

Of course, because the performance of SMO and Naïve Bayes algorithms are surpassed by BayesNet, we show the meta-learning result based on BayesNet only in Table 5.9, and ignore

the results using SMO or Naïve Bayes as base algorithms of meta-learning.

	Features	Number of Features	Pres.,Freq. Or Other Weighting Methods	Meta Learning Without Discretization	Meta Learning With Discretization	Difference
1	unigrams	1447	Pres.	60.2086	**61.0349**	0.8263
2		1447	Freq.	61.7358	**62.8003**	1.0645
3		1447	tf/idf	60.9962	**61.913**	0.9168
4		1447	Frequency_POS	60.0103	**60.4124**	0.4021
Avg				60.7377	61.5402	0.8025
5	Feature set 1 (and its variations)	1447	Pres.	58.2577	**59.0784**	0.8027
6		1447	Freq.	59.1335	**60.4458**	1.3123
7		1447	tf/idf	59.4346	**60.0154**	0.5808
8		1447	Frequency*SO(w)	58.1933	**58.7653**	0.572
Avg				58.7548	59.5762	0.817
9	Feature set 2 (and its variations)	1447	Pres.	62.9961	**64.7126**	1.7165
10		1447	Freq.	63.1244	**64.0749**	0.9505
11		1447	tf/idf	63.2332	**65.9755**	2.7423
12		1447	Frequency*SO(w)	63.5121	**65.6186**	2.1065
Avg				63.2165	65.0954	1.879
13	Feature set 3 (and its variations)	1451	Pres.	63.5675	**65.6493**	2.0818
14		1451	Freq.	63.9306	**65.3215**	1.3909
15		1451	tf/idf	63.8430	**65.3579**	1.5149
16		1451	Frequency*SO(w)	63.3524	**64.0327**	0.6803
Avg				63.6734	65.0904	1.417
Average improvement among all four types of Feature Sets						1.23

Table 5.9 Comparison of detailed accuracies of meta-learning between classifiers with and without discritization.

(Based on 1447 unigrams of word presence)

In our experiments, we used default parameters of discretization provided by WEKA. We specified the range of attributes to act on with the whole scope of attributes, i.e. with "first" and "last" valid values. This option made the discretization more equable. We did not change the parameters of 'invertSelection', 'makeBinary' 'useBetterEncoding', 'useKononenko' either, and left them as the initial default value'false'.

The improvement of accuracies is not large but remarkable. As we can see from the bottom line of Table 5.9, the average increase of accuracies of all four types of feature sets is 1.23%. This result is consistent with our expectation. Therefore we conclude that discretization is effective

for improving meta-learning result of our multi-class classification.

5.3.2.4 Combining Supervised Learning and Unsupervised Learning

As discussed in section 4.5.3.2, on one hand, the unsupervised learning method is more natural to capture the sentiment contained in natural language and is not easily affected by the skewed bank review datasets. However, unsupervised learning does not take advantage of the quantity of the samples in majority classes, as supervised learning does; on the other hand, for supervised learning, it is difficult to avoid excessively biasing toward majority classes even though re-sampling to some extent can compensate for the negative effect of imbalanced data.

Hence, we combine the score prediction results of class 2, 3 and 4 of unsupervised learning and the prediction results of class 1 and 5 of supervised learning, as shown in Figure 4.9. Natually, we speculate that this combination can learn from strong points of both methods to offset their weaknesses of them.

As described in Figure 5.15, the key part of the combination is that after checking the correctness of unsupervised learning, for all misclassified reviews by unsupervised learning, we cover the instances with supervised learning results if and only if the supervised learning (meta-learning) classified it into class 1 or class 5. With this combination we improved the prediction accuracy significantly as shown in Table 5.10.

	Features	Number of Features	Pres., Freq. Or Other Weighting Methods	Meta-Learning Based on Bayes Net	Combined method	Difference Of Precisions
1	unigrams	1447	Pres.	61.0349	**63.7628**	2.7279
2		1447	Freq.	62.8003	**65.4902**	2.6899
3		1447	tf/idf	61.913	**63.4519**	1.5389
4		1447	Frequency_POS	60.4124	**61.2248**	0.8124
Avg				61.5402	63.4824	1.9422
5	Feature set 1 (and its variations)	1447	Pres.	59.0784	**61.2530**	2.1746
6		1447	Ferq.	60.4458	**63.7605**	3.3147
7		1447	tf/idf	60.0154	**63.4943**	3.4789
8		1447	Frequency*SO(w)	58.7653	**59.892**	1.1267

Avg				59.5762	62.1	2.5238
9	Feature set 2 (and its variations)	1447	Pres.	64.7126	**67.8802**	3.1676
10		1447	Ferq.	64.0749	**67.1104**	3.0355
11		1447	tf/idf	65.9755	**68.1316**	2.1561
12		1447	Frequency*SO(w)	65.6186	**68.1588**	2.5402
Avg				65.0954	67.8203	2.7249
13	Feature set 3 (and its variations)	1451	Pres.	65.6493	**69.2405**	3.5552
14		1451	Ferq.	65.3215	**70.9802**	5.6587
15		1451	tf/idf	65.3579	**66.9798**	1.6219
16		1451	Frequency*SO(w)	64.0327	**64.868**	0.8353
Avg				65.0904	68.0171	2.9178
Average improvement among all four types of Feature Sets						2.5272

Table 5.10 Combined learning results vs. Meta-Learning

Table 5.10 shows a significant improvement. From the column of 'Combined method', we find all prediction accuracies are higher than 60% except on Feature Set 1(line 8) about the features weighted by Frequency*$SO(w)$. Furthermore, among all four types of feature sets, the combined method obtains a 2.5272% increase. Especially, this mothod achieves the best performance 70.9802% on the frequency features of Feature Set 3.

In this experiment, we still force resampling and dicretization on meta-learning. From the results reported in Table 5.10, we can conclude that when dealing with imbalanced data for multiclass classification tasks, the unsupervised learning may be a good alternative compensational method to counteract the weakness of supervised learning which is apt to bias toward majority classes. Our experiments proved that it is an effective approach in multiclass sentiment classification.

For the meta-learner using combined supervised and unsupervised learner, the t-test results of WEKA are shown as follows (the meta_2 in the second column represents the combined supervised and unsupervised learning):

```
Dataset                  (1) meta.     | (2) meta_2.
----------------------------------------------------
sentiment_analysis       (200)   61.03 |   63.76 v
sentiment_analysis       (200)   62.80 |   65.49 v
sentiment_analysis       (200)   61.91 |   63.45 v
sentiment_analysis       (200)   60.41 |   61.22
sentiment_analysis       (200)   59.08 |   61.25 v
```

```
sentiment_analysis        (200)   60.45 |   63.76 v
sentiment_analysis        (200)   60.02 |   63.49 v
sentiment_analysis        (200)   58.77 |   59.89
sentiment_analysis        (200)   64.71 |   67.88 v
sentiment_analysis        (200)   64.07 |   67.11 v
sentiment_analysis        (200)   65.98 |   68.13 v
sentiment_analysis        (200)   65.62 |   68.16 v
sentiment_analysis        (200)   65.65 |   69.24 v
sentiment_analysis        (200)   65.32 |   70.98 v
sentiment_analysis        (200)   65.36 |   66.98
sentiment_analysis        (200)   64.03 |   64.87
------------------------------------------------
                                  (v/ /*) |   (12/4/0)
```

There are four feature sets on which although the prediction accuracy is increased, the improvement is not statistically significant. These four feature sets are: the Unigrams of Frequency*SO(w), the Feature Set 1 of Frequency*SO(w), the Feature Set 3 of tf/idf, and the Feature Set 3 of Frequency*SO(w).

On the other hand, about all other 12 Feature Sets, the performance of the combined model is statistically better than the meta-learning method at the significance level of 0.05.

Therefore, we can conclude that the overall combined supervised and unsupervised learning statistically significantly better than the meta-classifiers.

It is worth noting that this method is not limited to our bank reviews data, because the reviews data in a wide range of domains on the website www.epinions.com present the same problem of imbalanced data distribution, including the domains of Automobile, Travel Destinations etc. Especially, the movie reviews are the most severly skewed domain of sentiment scoring. The combined algorithm, as a type of output engineering solution, is a promising method to improve the learning performance of this kind of multiclass classification.

Unfortunately, due to the time limitation and the complexity of invoving in new datasets, we did not run the combination method on a new dataset, and only using 10-fold cross validation test its performance based on the same dataset which is used by our unsupervised learning and

supervised learning.

Considering the cost of rebuild a set of experiments for new datasets, the time-consuming a serious procedures of review collection, data extraction, data cleansing and preprocessing, and the expensive learning process using WEKA, we leave this complete validation step in the future work. We do not further explore the validation result here in this thesis.

Chapter Six
Conclusions and Future Work

This thesis presents our heuristics and experiments regarding quantitative sentiment analysis. The purpose of our work is to determine the feasibility of quantitatively predicting the sentiment orientation of on-line documents. Furthermore, we perform basic research covering unsupervised learning and supervised learning based on 3164 customer reviews of 46 banks on www.epinions.com.

In this thesis, we explore the effect of various types of features such as unigrams, bigrams including valence shifters, SO-PMI-IR selected features, WordNet selected features, and WordNet derived synthetic semantic features; meanwhile, we experiment with different feature selection strategies, compare them, and discover the best measures for generating the most effective features. Furthermore, we implement a series of different learning algorithms on these feature sets, and adopt diversified approaches to improve the learning results. These approaches are cover most of the important phases of the machine learning process including input engineering, data cleansing, machine learning and output engineering.

Our testbeds are generated based on four feature sets: unigram feature set, Feature Set 1, Feature Set 2, and Feature Set 3. As described in chapter 5, we perform unsupervised learning on feature set 1 and feature set 2, and implement Naïve Bayes, BayesNet, C4.5 and SVM

algorithms on all the four feature sets.

At the beginning, our supervised learning experiments traverse all combinations of the four classifiers and the four feature sets, as shown in Table 5.3 of section 5.3.1.2. Afterwards, in the experiments of sections following section 5.3.1.2, we gradually remove ineffective classifiers and feature sets of low accuracies; and finally combine together the best unsupervised learning and supervised learning models to share their advantages.

In short, our experiments can be divided into three main parts:

- Incorporating WordNet measures with GI, SO-PMI-IR, and basic term counting methods, to explore the effects of WordNet selected features in quantitative sentiment analysis tasks, and comparing the performance between WordNet selected features and traditional features which are widely used in previous works.

- Implementing and improving basic unsupervised learning and supervised learning strategies on the four feature sets, and observing their effects and analyzing their advantages and disadvantages in quantitative sentiment analysis tasks.

- Experimenting with different strategies for improving learning performance, such as using re-sampling to alleviate the effect of imbalanced data, using discretization to increase the learning efficiency and accuracy, making use of potential order information in class label by meta-learning method, and combining multiple learning models.

These three parts provide sufficient arguments for analyzing the performance of various classifiers, introducing good measures for feature selection process, recommending promising learning algorithms and strategies, suggesting challenges and difficulties we need to pay attention to, and proposing the direction of future works.

In this chapter, we discuss our conclusions in section 6.1; and then suggest the crucial

problems of future works in section 6.2.

6.1 Conclusions

Through all the experiments in this thesis, we reached the following conclusions:

1. In both unsupervised learning (section 5.2) and supervised learning (section 5.3), the accuracies of predicting sentiment score of bank reviews are much higher than that of random-assignment baseline result (section 5.2.4). The predominant performance of the former method over the random-assignment method indicates that machine learning is a feasible and effective approach for quantitative sentiment analysis.

   Our unsupervised learning and supervised learning models achieve at highest 52.5215% (section 5.2.4, by unsupervised learning on Feature Set 2) and 70.9802% (section 5.3.2.4, by combined model on Feature Set 3) accuracy respectively; both results evidently surpass the setup baseline of 20.6384% and 20.8281% on Feature Set 1 and Feature Set 2.

   Because of the lack of references for multi-class sentiment classification or sentiment score prediction research, so far we have no idea how our learning performance compares to other similar tasks. The "Thumbs Up" or "Thumbs Down" sentiment analysis research by Peter D.Turney achieved 84.21% (averagely 80% to 84% on different specific domain of reviews) accuracy for binary classification, while it is not comparable with our sentiment analysis for multiclass classification. We feel that taking the accuracy from 52% to 71% represents important progress in the task of five-class sentiment classification.

   From another point of view, we also conclude that five star score prediction is a more challenging machine learning task than binary sentiment classification.

2. The combination of unsupervised and supervised learning methods significantly improves the performance of our multiclass classification task. Our experimental results show that

combining different learning models can benefit from the advantages and reduce the weaknesses of each participant model.

When talking about the principle of choosing learning schemes for real machine learning problems, there have long been controversy resulted from different philosophical point of views. Some people prefer simple theories over complex ones, but other people consider that if several schemes achieve similar accuracy , it may be possible to achieve a higher degree of precision by using them together.

The former opinion comes from the appreciation of *Occam's Razor*, an idea that the best scientific theory is the smallest one that explains all the facts. Actually, in our experiments, the better performance of BayesNet over SVM and C4.5 proves the validity of this 'simplicity' principle. Similarly, Naïve Bayes, the second best classifier which obtained competitive prediction accuracy with BayesNet in the multiclass classification, again agree with this principle.

However, the preference for simplicity should only be a scenario-specific principle rather than an absolute rule. This opinion is based on Epicurus's *principle of multiple explanations,* which advises "if more than one theory is consistent with the data, keep them all". This principle brings to mind *bagging, boosting, stacking and error-correcting codes* where the output of several different models are combined together to make the decisions more reliable.

Bagging, boosting, and stacking are general techniques that can be applied to numric prediction problems as well as classification tasks. However, our sentiment scoring is treated as multiclass classification task, so above three methods are not appropriate to our case. (Note: We do not exclude the possibility of considering score prediction to be numeric prediction task. Actually, many customer review websites support scoring products or services at 0.5 even 0.1 interval. In this way, sentiment score prediction becomes more and more like a machine learning task between multiclass classification and regression problems. The extreme case of numeric scoring is the prediction of continuous

value. Although this area is out of the scope of our thesis, we still think these three methods are promising candidates for solving continuous prediction problems.)

The error correcting codes method is less general than the other three techniques mentioned above because it applies to classification problems, and happens to be used to ones that have more than two classes. Unfortunately, however, in the preliminary study, we have tried error-correcting codes, but it caused the performance of BayesNet and Naïve Bayes both to decline severely. Therefore, we exclude it from the potential solutions.

When solving the imbalanced classification problem by re-sampling, we accidentally discovered the different class distribution between the outputs of unsupervised learning and supervised learning results. Because unsupervised learning does not bias toward majority classes, we combine the prediction result of minority classes of unsupervised learning with that of the majority classes of supervised learning to overcome the serious bias of classification, and then, obtain the best accuracy as high as 70.9802%. (There is another part of the complete validation tests are left to the future work.)

This result indicates that in addition to, combining the output of several different models is a feasible and reasonable choice especially when the data presents biased distribution tendency, because this approach may significantly conquer the weakness resulting from skewed data or other reasons. The key advantage of unsupervised learning is the nature that it is more apt to capture the sentiment of natural language in the way of its intrinsic attribute, without the bias toward majority classes.

3. For multiclass sentiment classification, unsupervised learning and weak supervised learning algorithms can produce competitive performance comparable to that of sophisticated and complicated supervised learning methods.

In our experiments, without any improvement or optimization, the supervised learning does not overwhelmingly surpass unsupervised learning algorithms. As shown in Table 5.2, the unsupervised learning achieves 52.5215%, and the supervised classifiers obtain

58.4503% accuracy at highest, the difference is not as prominent as what we expect initially.

On the other hand, it is surprising that the prediction accuracy of C4.5 is not only lower than that of BayesNet and Naïve Bayes, but also far worse than the result of unsupervised learning. In Table 5.3, the average accuracy of C4.5 is only 46.5124%, and it is outperformed by both of the accuracy of unsupervised learning (52.5215%) and average accuracies of BayesNet (57.5951%).

The superiority of our unsupervised learning over the influential decision tree C4.5 is attributed to their respective nature. Compared to C4.5 decision tree, the simple probabilistic method of our unsupervised learning is more natural and straightforward, and more apt to successfully capture the potential ordinal ranking of the score of customer reviews. This is in accordance with the *Occam's Razor* principle we mentioned above in that the best scientific theory usually is the smallest one that explains all the facts, while our unsupervised learning just is the small and simple method, but it is not simpler. It is simple but reasonable.

C4.5 may be good at making a decisions between two categories of customer reviews that clearly belong to "Thumbs up" or "Thumbs Down", but it is not appropriate to rank an entire corpus in the order of the their degree of subjectivity. Moreover, the regular C4.5 decision tree model is not adept at dealing with the particular problem of imbalanced data, because the skewed data usually results in bias towards the majority of decision tree structure, and this phenomenon is closely related to its "divide and conquer" working mechanism.

According to our understanding, following reasons caused the bad performance of C4.5:

a) Decision trees are more suited to nominal attributes, but the features in our experiments are all real values. During the construction of the decision tree, a nominal attribute can only be tested once on any path from the root of a tree to the

leaf, but a numeric one can be tested many times. This process yields trees that are messy and difficult to understand. In other words, the numeric value of features in our dataset may cause the tree to grow to an excessively large and unreasonable size.

b) The bad performance could closely relate to the pruning process of decision tree. Originally, the pruning operation is used to control the possible overfitting phenomenon of decision tree classifiers. However, it does not help overcome the overfitting in our experiments:

First, the imbalanced data of bank reviews (see table 4.10 in section 4.5.3.1) inherently results in the overfitting, i.e. the bias towards majority classes. This kind of overfiting cannot be solved by pruning.

Secondly, the pruning operation may hyper-correct the decision tree based on our dataset. C4.5 decision tree adopt the subtree raising strategy to do pruning. Its principle is sacrificing the accuracy of the tree built on the training dataset, to improve the performance on the test dataset. However, because all of the attributes in our dataset are numeric and our task is multi-class classification, the subtree raising is apt to excessively prune useful branches.

Furthermore, in actual implementations, the subtree raising process is generally restricted to raising the subtree of the most popular branch. This inclination further deteriorates the situation especially under our imbalanced class distribution.

Consequently, in our experiments, the development process of C4.5 decision tree is apt to become inappropriate and unreasonable.

c) Because of the specialty of the numeric attributes and the multiclass classification task, the subtree raising process becomes very time-consuming (over 60 hour's runtime for J48 in WEKA), so we have to think about the discretization.

Decision tree classifiers deal with numeric attributes on a local basis. When it is being constructed, the algorithm examines attributes at each node of the tree to check if they are worth branching on. On the other hand, local discretization is tailored to the actual context provided by each tree node and will produce different discretization of same attribute at different positions in the tree if that seems reasonable.

However, its decisions are based on less data as tree depth increases, which compromises their reliability. With the normal technique of pruning, it is obvious that many discretization decisions will be based on data that is grossly inadequate. This phenomenon happens in our experiments, and results in the abnormal situation shown in Figure 6.1:

```
=== Detailed Accuracy By Class ===

TP Rate   FP Rate   Precision   Recall  F-Measure   ROC Area  Class
  0.852     0.453      0.61      0.852     0.711      0.781    1.0
  0.113     0.102      0.153     0.113     0.13       0.525    2.0
  0         0          0         0         0          0.618    3.0
  0.015     0.007      0.231     0.015     0.028      0.604    4.0
  0.616     0.168      0.467     0.616     0.531      0.812    5.0

=== Confusion Matrix ===

    a    b    c    d    e   <-- classified as
 1224  100    0    7  105 |    a = 1.0
  328   50    0    2   62 |    b = 2.0
  157   40    0    2   69 |    c = 3.0
  150   59    0    6  193 |    d = 4.0
  147   78    0    9  376 |    e = 5.0
```

Fig 6.1 The learning result of C4.5 with discretization

Obviously, due to the inadequate data during discretization and initially imbalanced data distribution, C4.5 classifier classifies no instances into the minority class of score 3. Consequently, C4.5 leaves a dilemma between saving time and preveting the deterioration of its accuracy to us.

Finally, the C4.5 decision tree obtains almost all of the lowest prediction accuracies in our experiments in chapter 5. Our observation is consistent with previous works by Michael Gamon & Anthony Aue [17], Andrew Lacey [20] and Peter D.Turney [14].

Therefore, we conclude that in the sentiment classification task, due to its domain-specific character, unsupervised learning and weak supervised learning should not be excluded; on the contrary, these unsupervised learning and weak supervised learning algorithms are promising candidates. Furthermore, we delete C4.5 from our list of candidate classifiers since section 5.3.2.

4. Incorporating bigrams with unigrams together, either capturing valence shifters or not, does not necessarily improve the performance of multiclass sentiment classification. In fact, whether the sentiment learning process benefits from bigrams relies on the quality of unigram and bigrams features.

   Feature Set 1 in our experiments comprises unigrams and bigrams selected by GI and SO-PMI-IR algorithms, but it always outperformed by the pure unigram feature set. This result is ascribed to the inappropriate feature selection strategy. After using WordNet measures for feature selection process and adding synthetical semantic features, the multi-class classification performance surpasses the results based on unigram feature set and Feature Set 1.

5. As discussed above, the WordNet selected features benefit supervised learning greatly. As shown in Table 5.3, in section 5.3 and in following experiments, the classification benefits more from features which are selected using SO_WN algorithm than those extracted by the SO-PMI-IR algorithm only. Based on Feature Set 2 and Feature Set 3, the performance of BayesNet, Naïve Bayes, and SVM all surpass their performance on the unigram feature set and Feature Set 1.

   The effect of WordNet selected features indicates the possible interaction between

subjectivity and word sense. In other words, a close correlation exists between these two semantic properties of natural language. Therefore, using WordNet to select features is a beneficial complement to traditional approaches of sentiment feature selection.

6. The re-sampling greatly improves the performance of imbalanced multiclass classification. In our scenario, we use all default parameters provided by WEKA. We do not use either sampling with replacement or the 'bias towards a uniform one' option. In other words, we use re-sampling with parameters that minimize the overfitting possibility, and still obtain significant improvement. Therefore, we can conclude that re-sampling is a very effective method to deal with skewed data, expecially in multiclass classification.

7. The ordinal meta-learning method successfully makes use of the ordering information in the class attribute.

   In section 5.3.2.1, we present the result of the ordinal meta-learning method that makes use of ordering information in ordinal class labels. The method converts the original multiclass sentiment classification problem into a series of binary class problems that encode the ordering information of the original classes.

   Our empirical experiments in section 5.3.2.1 show that the ordinal meta-learning method brings a 2.96% increase of accuracy to BayesNet algorithm, and makes it achieve 60.5549% accuracy. Similarly, this approach also contributes an increase of 1.91% for Naïve Bayes classifier. (Note: The abnormal result on SVM algorithm is an exception relates to the the nature of SVM kernel; in other words, this meta-learning is not appropriate to SVM based classifier, for the detailed discussion please refer to section 5.3.2.1)

   Our observation demonstrates that a significant improvement in performance is achieved by exploiting ordering information. In short, our experiments prove that meta-learning is a successful and effective method for solving the kinds of problems found in multiclass sentiment classification tasks.

6.2 Future Work

1. For the promising combination of unsupervised learning and supervised learning methods, we need to test it on a different dataset to validate its effectiveness, and accumulate more experience of how to adjust both methods and the combination strategy to the real data distribution.

   Due to the time limitations and the cost of implementing new experiments, we did not complete the experiments on a completely new dataset. It will be very useful and helpful to experiment with the combined models on a fresh dataset.

2. Our experiments are based on the bank reviews, and obtain very good performance by different learning methods on various feature sets. In the whole process of the experiments, we get significant improvement when facing different difficulties and problems.

   The good perfromance of our experiments might be attributed to the nature of the customer reviews of banks. In other words, it is hard to exclude the possibility that the results take advantage of a very straightforward and unembellished writing style.

   Therefore, we expect to implement our methods of this thesis on the movie review dataset which usually include more implicit expressions than bank reviews, and explore how our algorithms cope with some more challenging problems such as the "thwarted narrations" (see Bo Pang [10]).

3. As mentioned in section 6.1, many customer review websites support scoring products or services at 0.5 or even 0.1 interval. In this way, sentiment score prediction becomes more and more like a task between multiclass classification and regression problems.

   We would like to solve the sentiment classification problem by linear or non-linear regression algorithms; concurrently, we can use combined models of regression method with other basic machine learning approaches such as Decision Tree, BayesNet, SVM etc.

(for example, use *classification and regression trees* (CART) and so on) to share all of their advantages.

Moreover, some combining models including bagging, boosting, and stacking which are applied to the numeric prediction problems, will be available if we adopt a regression algorithm to deal with sentiment classification, and treat it as a numeric prediction task. And then, by approximating the regression result to integer the values from 1 to 5, it is easy to obtain the result of five star scoring.

4. When using re-sampling to solve the problem of imbalanced data, there are a lot of candidate methods worth trying. For example, the SMOTE [68] method by N. Chawla, A. Lazarevic, L. Hall, K. Bowyer solved the dilemma of oversampling and overfiting very well.

   Due to time limitations, we do not delve into re-sampling as deeply as we would like, but in future we would spend more time on delicately and gradually improving the re-sampling method, because there is a lot of room for improvement.

5. It is worth trying sentence level sentiment recognition.

   In this thesis, we do not pay much attention to the sentiment classification at the sentence level, but it may improve the performance of our sentiment classification task on bank reviews. In addition, this is an effective approach to decrease the learning cost.

   Wiebe et al [16] have already issued a method of separating the subjective sentences from the objective ones. Based on this preliminary step, we can narrow down the scope from which we extract the sentiment features. Given the split subjective sentences, we can focus the feature selection on these subjective sentences only. We will incorporate this step with our preprocessing process in the future.

6. There has always been an occasional thought that, to some extent, the five star rating result

may not necessarily hundred percent correspond to the authors' review content, because reviewers' expression depends on their writing skills, their mood during the writing, and their writing habits etc. Originally, there may be a difference between customers' reviews and their ratings.

Therefore, first of all, we need create an evaluation system to accurately evaluate and indicate the relationship between the written reviews and their scores; otherwise, we cannot accurately evaluate the performance of our machine learning methods either.

To conclude, we believe that using the methods we explore in this thesis, and via continuous improvement, quantitative sentiment analysis can achieve the performance that is comparable to the accuracy of previous binary sentiment classification using more sophisticated and complicated algorithms than ours.

Bibliography

[1] K. Dave, S. Lawrence, and D. M. Pennock. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proc. of the 12th Int. WWW Conf.*,2003.

[2] Christopher D. Manning, and Hinrich Schiitze. Foundations of statistical natural language processing. 1999 Massachusetts Institute of Technology Second printing with corrections, 2000

[3] D. D. Lewis. Naive (Bayes) at forty: The Independence Assumption in Information Retrieval. In Proceedings of the 10th European Conference on Machine Learning, New York, 1998, 4-15.

[4] S. Eyheramendy, D. D. Lewis and and D. Madigan. On the naive bayes model for text categorization. Artificial Intelligence&Statistics 2003.

[5] F. Peng and D. Schuurmans. Combining naive bayes and n-gram language models for text classification. Proceedings of The 25th European Conference on Information Retrieval Research (ECIR03). April 14-16, 2003, Pisa, Italy.

[6] Y Yang. An evaluation of statistical approaches to text categorization. Information Retrieval, 1999, 1(1):7688.

[7] Y Yang and C.G. Chute. A linear least squares fit mapping method for information retrieval from natural language texts. In Proceedings of the 14th Conference on Computational Linguistics (COLING92), 1992.

[8] T. Joachims, T. Mitchell, D. Freitag, and R. Armstrong. Webwatcher: machine learning and hypertext. In K. Morik and J. Herrmann, editors, GI FachgruppentrefFen Maschinelles Lernen, University of Dortmund, August, 1995.

[9] C. Hsu, C. Lin. A comparison on methods for multi-class support vector machines,IEEE Transactions on Neural Networks. 2002, 13: 415425.

[10] Pang, Bo, Lillian Lee and Shivakumar Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79-86.

[11] Alistair Kennedy and Diana Inkpen, 2006. Sentiment Classification of Movie Reviews Using Contextual Valence Shifters. Computational Intelligence Volume 22 Page 110 - May 2006 doi:10.1111/j.1467-8640.2006.00277.x

[12] Yi, J. Nasukawa, T. Bunescu, R. Niblack, W. Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques. Data Mining, 2003. ICDM 2003. Third IEEE International Conference on Publication Date: 19-22 Nov. 2003 pages: 427- 434

[13] Hatzivassiloglou, V. and Mckeown, K. 1997. Predicting the semantic orientation of adjectives. *In* Proceedings of the 35th Annual Meeting of the Association for Computational Linguistics (ACL'97), pp. 174–181, Madrid, Spain.

[14] Turney, P. 2002. Thumbs up or thumbs down? Semantic orientation applied to unsupervised classification of reviews. *In* Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL 2002), pp. 417–424, Philadelphia, PA.

[15] Tetsuya Nasukawa and Jeonghee Yi. 2003. Sentiment analysis: Capturing favorability using natural language processing. In *Second International Conference on Knowledge Capture*,Florida, USA.

[16] Vasileios Hatzivassiloglou, Janyce M. Wiebe. Effects of adjective orientation and gradability on sentence subjectivity. Pages: 299 – 305. Proceedings of the 18th conference on Computational linguistics - Volume 1 July 2000

[17] Michael Gamon, Anthony Aue. Automatic identification of sentiment vocabulary: exploiting low association with known sentiment terms. Proceedings of the ACL 2004 Workshop on Feature Engineering in Natural Language Processing. Page: 57-64 June,2005

[18] Mullen, Tony and Nigel Collier. 2004. Sentiment analysis using Support Vector Machines with diverse information sources. In Dekang Lin and Dekai Wu, editors, *Proceedings of EMNLP 2004*, pages 412-418, Barcelona, Spain. Association for Computational Linguistics.

[19] Mullen, Tony and Nigel Collier. Incorporating topic information into sentiment analysis models. National Institute of Informatics (NII). 2003

[20] Andrew Lacey. A Simple Probabilistic Approach to Ranking Documents by Sentiment. *Proceedings of the Class of 2005 Senior Conference*, pages 1–7

[21] Peter D. Turney and Michael L. Littman.2002. Un-supervised learning of semantic orientation from a hundred-billion-word corpus. Technical Report EGB-1094, National Research Council Canada.

[22] J. Wiebe, R. Bruce, and T. O'Hara. 1999. Development and use of a gold standard data set for subjectivity classilieations. In *Proceedings of tile 37th Antotal Meeting of the Association*

for Computational Li,guistics (ACL-99), pages 246-253

[23] Rebecca Bruce and ,lanyce Wiebe. 2000. Recognizing subjectivity: A case study of rllanual tagging. *Natural Language E, gineering,* 6(2).

[24] Passonneau, R. and D. Litman. 1993. Intention-based segmentation: Human reliability and correlation with linguistic clues. In Proc. 31st Annual Meeting of the Association for Computational Linguistics (ACL-93), pages 148{155. Association for Computational Linguistics.

[25] J. Wiebe, T. Wilson, and M. Bell. 2001. Identifying collocations for recognizing opinions. In *Proc. ACL-01Workshop on Collocation: Computational Extraction,Analysis, and Exploitation*, July.

[26] Theresa Wilson, David R. Pierce, Janyce Wiebe 2003. Identifying Opinionated Sentences. Proceedings of HLT-NAACL 2003 Demonstrations , pp. 33-34

[27] Epinions, http://www.epinions.com

[28] H. Schutze. 1998. Automatic word sense discrimination.*Computational Linguistics*, 24(1):97–123.

[29] D. Inkpen and G. Hirst. 2003. Automatic sense disambiguation of the near-synonyms in a dictionary entry. In *Proceedings of the 4th Conference on Intelligent Text Processing and Computational Linguistics(CICLing-2003)*, pages 258–267, Mexico City, February.

[30] Satanjeev Banerjee and Ted Pedersen. 2002. An adapted Lesk algorithm for word sense disambiguation using WordNet. In *Proceedings of the Third International Conference on Intelligent Text Processing and Computational Linguistics(CICLing-02)*, Mexico City.

[31] M. Lesk. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from a ice cream cone. In *Proceedings of SIGDOC'86.*

[32] S. Patwardhan and T. Pedersen. Using WordNet-based context vectors to estimate the semantic relatedness of concepts. In Proceedings of the REFERENCES 47 EACL 2006 Workshop on Making Sense of Sense: Bringing Computational Linguistics and Psycholinguistics Together, Trento, Italy, April 2006.

[33] http://www.imsc.res.in/~golam/gposttl/

[34] Lyons, J. 1977. Semantics. 2 vols. New York: Cambridge University Press.

[35] http://www.dcs.gla.ac.uk/idom/ir_resources/linguistic_utils/stop_words

[36] http://www.perl.com/2003/02/19/examples/VectorSpace.pm.

[37] Hirst G., St-Onge D. 1997. Lexical Chains as representation of context for the detection and correction malapropisms. In C. Fellbaum, editor, WordNet: An electronic lexical database and some of its applications. Cambrige, MA: The MIT Press.

[38] M. Hearst. Direction-based text interpretation as an information access refinement. Text-Based Intelligent Systems, 1992.

[39]W. Sack. On the computation of point of view. In Proc. Of the 12th AAAI Conf., 1994.

[40] S. Das and M. Chen. Yahoo! for anazon: Extracting market sentiment from stock message boards. In Proc. of the 8th APFA, 2001.

[41] R. M. Tong. An operational system for detecting and tracking opinions in on-line discussion. In SIGIR Workshop on Operational Text Classification, 2001.

[42] P. Subasic and A. Huettner. Affect analysis of text using fuzzy semantic typing. IEEE Trans. on Fuzzy Systems, Special Issue, Aug., 2001

[43] C. Whissell. The dictionary of affect in language. Emotion: Theory, Research, and Experience, pages 113–131.

[44] H. Li and K. Yamanishi. Mining from open answers in questionnaire data. In Proc. of the 7th ACMSIGKDD Conf., 2001.

[45] W. Sack. On the computation of point of view. In Proc. Of the 12th AAAI Conf., 1994.

[46] S. Morinaga, K. Yamanishi, K. Teteishi, and T. Fukushima. Mining product reputations on the web. In Proc. of the 8th ACM SIGKDD Conf., 2002.

[47] L. Terveen,W. Hill, B. Amento, D. ḾcDonald, and J. Creter. PHOAKS: A system for sharing recommendations. CACM,40(3):59–62, 1997.

[48] Church, K.W., & Hanks, P. 1989. Word association norms, mutual information and lexicography. Proceedings of the 27th Annual Conference of the ACL (pp. 76-83). New Brunswick, NJ: ACL.

[49] Brill, E. 1994. Some advances in transformation-based part of speech tagging. Proceedings of the Twelfth National Conference on Artificial Intelligence (pp.722-727). Menlo Park, CA: AAAI Press.

[50] G. Mishne. Experiments with mood classification in blog posts. In Style2005 – 1st Workshop on Stylistic Analysis of Text for Information Access, at SIGIR 2005, 2005.

[51] Frank, E. & Hall M. (2001). A simple approach to ordinal classification. (Working paper series. University of Waikato, Department of Computer Science. No. 01/5/2001). Hamilton,

New Zealand: University of Waikato.

[52] Davy Temperley, Daniel Sleator, and John Lafferty, Link Grammar Parser, www.link.cs.cmu.edu/link/

[53] R. Bruce and J. Wiebe 2000, Recognizing subjectivity: A case study of manual tagging Natural Language Engineering, 6(2).

[54] Alison Huettner and Pero Subasic. 2000. Fuzzy typing for document management. In ACL 2000 Companion Volume: Tutorial Abstracts and Demonstration Notes, pages 26–27.

[55] Sanjiv Das and Mike Chen. 2001. Yahoo! For Amazon: Extracting market sentiment from stock message boards. In Proc. of the 8th Asia Pacific Finance Association Annual Conference (APFA 2001).

[56] D. McCarthy, R. Koeling, J. Weeds, and J. Carroll. 2004. Finding predominant senses in untagged text.In Proc. ACL 2004.

[57] J. Jiang and D. Conrath. 1997. Semantic similarity based on corpus statistics and lexical taxonomy. In Proceedings on International Conference on Research in ComputationalLinguistics, pages 19–33, Taiwan.

[58] S. Banerjee and T. Pedersen. 2003. Extended gloss overlaps as a measure of semantic relatedness. In Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, pages 805–810, Acapulco, August.

[59] Ian H. Witten and Eibe Frank. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementation.* Morgan Kaufmann, San Francisco, 2000.

[60] Weiss, G.M. and Provost, F. (2003) "Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction", Volume 19, pages 315-354.

[61] N. Japkowicz (2000). The Class Imbalance Problem: Significance and Strategies, In the Proceedings of the 2000 International Conference on Artificial Intelligence(ICAI'2000): Special Track on Inductive Learning.

[62] N. Chawla, A. Lazarevic, L. Hall, K. Bowyer (2003). SMOTEBoost: Improving Prediction of the Minority Class in Boosting, 7th European Conference on Principles and Practice of Knowledge Discovery in Databases, Cavtat-Dubrovnik, Croatia, 107-119.

[63] Trevor Hastie, Robert Tibshirani: Classification by Pairwise Coupling. In: Advances in Neural Information Processing Systems, 1998.

[64] Jaap Kamps, Maarten Marx, Robert J. Mokken, and Marten de Rijke. 2002. Words with

attitude. *In In Proceedings of the 1st International Conference on Global WordNet*, Mysore, India.

[65] Charles E. Osgood, George J. Succi, and Percy H. Tannenbaum. 1957. *The Measurement of Meaning*.University of Illinois.

[66] Z. Wu and M. Palmer. 1994. *Verb semantics and lexical selection.* In 32nd Annual Meeting of the Association for Computational Linguistics, pages 133–138, Las Cruces, New Mexico.

[67] POLANYI, L. and ZAENEN, A. 2004. *Contextual valence shifters.* In J. Shanahan, Y. Qu, and J. Wiebe (eds.), Computing Attitude and Affect in Text: Theory and Applications, pp. 1–9. The Information Retrieval Series, Vol. 20, Springer, Dordrecht, The Netherlands.

[68] Janyce Wiebe and Rada Mihalcea . 2006. *Word sense and subjectivity.* Annual Meeting of the ACL Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the ACL Sydney, Australia Pages: 1065 – 1072, Year of Publication: 2006

[69] RILOFF, E. and WIEBE, J. 2003. *Learning extraction patterns for subjective expressions.* In Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, pp. 105–112.

[70] WIEBE, J., WILSON, T., BRUCE, R., BELL, M., and MARTIN, M. 2004. *Learning* subjective language. Computational Linguistics 30(3) pp. 277–308.

[71] PANG, B. and LEE, L. 2004. *A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts.* In Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), pp. 271–278, Barcelona, Spain.

[72] YU, H. and HATZIVASSILOGLOU, V. 2003. *Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences.* In Proceedings of the 2003 Conference on Empirical Methods in Natural Language Processing, pp. 129–136.

[73] Evgeniy Gabrilovich and Shaul Markovitch . *Text Categorization with Many Redundant Features: Using Aggressive Feature Selection to Make SVMs Competitive with C4.5.* The 21st International Conference on Machine Learning (ICML) , pp. 321-328, Banff, Alberta, Canada, July 2004.

[74] Thomas Bayes (1763). "An Essay towards solving a Problem in the Doctrine of Chances. By the late Rev. Mr. Bayes, F.R.S., communicated by Mr. Price, in a letter to John Canton, A.M., F.R.S.". Philosophical Transactions of the Royal Society of London 53: 370–418.

[75] Bo Pang and Lillian Lee. *Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales.* Proceedings of ACL 2005, pp. 115&ndash124.

[76] Michael Gamon 2005. *Sentiment Detection and its Applications.* In Proceedings of the ACL-05 Workshop on Feature Engineering

Appendix

Appendix A – The list of 46 banks and the number of reviews

The Name of Banks	Regular Reviews	Express Reviews	Total Number
AmSouth	22	4	26
AmTrust	9	2	11
Astoria_Federal_Savings_and_Loan_Association	12	1	13
BankBoston	38	0	38
Bank_of_America	463	36	499
Bank_of_New_York	10	0	10
Bank_of_the_West	10	4	14
Bank_One	113	5	118
Branch_Banking_and_Trust_Company	18	1	19
California_Federal_Bank	18	0	18
Centura_Bank	16	2	18
Charter_One_Bank	44	3	47
Chase	85	13	98
Chevy_Chase_Bank	25	3	28
Citibank	120	14	134
Citizens_Bank_-_Rhode_Island	23	4	27
Commerce_Bank_(NY-NJ-PA-DE)	20	17	37
Commerce_Bank_-_Missouri	27	0	27
Compass_Bank_-_Alabama	35	4	39
Emigrant_Savings_Bank	14	11	25
Fifth_Third_Bancorp	37	6	43
Firstar	33	0	33
First_Security	11	0	11
First_Union	135	0	135
Fleet_Financial_Group	92	2	94
HSBC_Bank_USA	45	15	60
ING_Direct	120	75	195
KeyCorp	48	5	53
M&T_Bank	34	6	40
Mellon_Bank	16	0	16
National_City	54	2	56
PNC_Bank	41	9	50
Provident_Bank_of_Maryland	14	1	15
Regions_Financial	21	1	22
SouthTrust_Bank	17	0	17
Sovereign_Bank	40	6	46
SunTrust_Bank	44	6	50
TCF_National_Bank_Illinois	14	1	15
TCF_National_Bank_Minnesota	16	4	20
U.S._Bancorp	70	3	73
Union_Bank	17	5	22
Union_Planters	16	1	17
USAA	70	10	80

Wachovia	80	13	93
Washington_Mutual	282	41	323
Wells_Fargo	314	25	339
Total Numbers of All Banks	2803	361	3164

Appendix B – The mapping between Penn tags from Reviews and the POS of noun, verb, adjective and adverb in WordNet.

Penn Tags from Reviews	POS in WordNet
PDT	ADJECTIVE
AT	
RBR	ADVERB
CD	ADJECTIVE
RP	ADVERB
VHP	VERB
EX	
RB	ADVERB
FW	
VVG	VERB
VBP	VERB
NPS	NOUN
JJR	ADJECTIVE
UH	
VVZ	VERB
JJS	ADJECTIVE
VB	VERB
MD	VERB
NNS	NOUN
VVD	VERB
NP	NOUN
RBS	ADVERB
PP	
VV	VERB
VBG	VERB
VH	VERB
CC	
VBN	VERB
WP	
JJ	ADJECTIVE
TO	
WRB	ADVERB
VHG	VERB
WDT	
VVN	VERB
VBZ	VERB
VHD	VERB
NN	NOUN
VBD	VERB
DT	
VHN	VERB
IN	
VHZ	VERB
VVP	VERB