Decision Boundary Setting and Classifier Combination for Text Classification

Moch Arif Bijaksana

A Thesis submitted for the degree of Doctor of Philosophy

Science and Engineering Faculty

Queensland University Of Technology

March 2015

Abstract

Text classification is a popular and important text mining task. Many document collections are multi-class and some are multi-label. Both multi-class and multi-label data collections can be dealt with by using binary classifications. A big challenge for text classification is the noisy text data. This problem becomes more severe in corpus with small set of training documents, moreover accompanied by few positive documents. A set of natural language text contains a lot of words. This results another important problem for text classification, namely, high dimension data. Therefore we must select features. A classifier must identify boundary between classes optimally. However, after the features are selected, the boundary is still unclear with regard to mixed positive and negative documents. Recently, relevance feature discovery (RFD) has been proposed as an effective pattern mining-based feature selection and weighting model. Document weights are significant for ranking relevant information. However, so far, an effective way to set the decision boundary for ranking relevant information for classification has not found. This thesis presents a promising boundary setting method for solving this challenging issue to produce an effective text classifier, called RFD$_\tau$. A classifier combination to boost effectiveness of the RFD$_\tau$ model is also presented. The experiments carried out in the study demonstrate that the proposed classifier significantly outperforms existing, including state of the art, classifiers.

Contents

List of Figures

List of Tables

Declaration of Authorship

The work contained in this thesis has not been previously submitted to meet requirements for an award at this or any other higher education institution. To the best of my knowledge and belief, the thesis contains no material previously published or written by another person except where due reference is made.

QUT Verified
Signature

Signed : ____________________

Date : 12 March 2015

Acknowledgements

Praise and thanks to Allah for all His mercy and compassion, because this PhD journey would never have finished successfully without His blessings.

I would like to express my sincere gratitude to my principal supervisor Professor Yuefeng Li who has supported me throughout my thesis with his patience and knowledge.

I would also like to thank my associate supervisor Dr. Laurianne Sitbon for her encouragement and support.

I would also like to thank my fellow labmates in e-Discovery Lab for their collaboration, encouragement and friendship.

Also many thanks to Helen Whittle for the thesis proofreading.

Finally, I must express my very profound gratitude to my beloved family who patiently helped me to finish this project, my wife Atik and my children Ika, Ina and Naufal.

Chapter 1

Introduction

1.1 Background

In the age of the internet, people and organisations face more and more information. Text mining, which is the automatic extraction of implicit and potentially useful information from text, has therefore become increasingly important. Several important techniques in text mining include clustering, classification, and association mining. Text classification is used in many areas such as the filtering of unwanted information (spam web pages, spam email), the filtering of specific information (information filtering), organising personal email, sentiment detection (automatic classification of a movie or product review as positive or negative), and vertical searching (searches on a specific topic) [105].

The text classification task is to assign a Boolean value to each pair $\langle d_j, c_i \rangle \in \mathcal{D} \times \mathcal{C}$ where $\mathcal{D}$ is a domain of documents and $\mathcal{C} = \{c_1, \ldots, c_{\mathcal{C}}\}$ is a set of predefined classes/categories. The task is to approximate the true function $\Phi \colon \mathcal{D} \times \mathcal{C} \to \{1, 0\}$ by means of a function $\breve{\Phi} = \mathcal{D} \times \mathcal{C} \to \{1, 0\}$, such that $\breve{\Phi}$ and Φ 'coincide as much as possible'. The function Φ is called a classifier, and the coincidence is the effectiveness of the classifier [51, 144].

A text classification system normally includes three components, namely, initial preprocessing, document representation, and classification models [144]. In the initial preprocessing, which involves parsing, stemming, cleaning, and stop word removal, standard methods usually provide satisfactory classification performance. In document representation, the important issue in preprocessing is feature weighting and selection. A lot of research has been conducted on the term weighting and selection problem. Different classification models are used in the classifiers, such as support vectors in support vector machines (SVM), centroids in Rocchio, probability in Naive Bayes, and tree in C4.5 [144].

In real life, many classification problems are multi-class and multi-label. A multi-class dataset has two or more classes, and each document has one class. In a multi-label dataset, one or more classes can be assigned to a document. A dataset can be both multi-class and multi-label. Problem of multi-class and multi-label classification is commonly solved by splitting into several single-label binary-class (or in short, binary class) classification problems [76]. The binary classification is a special multi-class with two classes, i.e., $\mathcal{C} = \{c_1, c_2\}$. A binary classification is theoretically more generic than the multi-class classification or multi-label classification [144]. This thesis focuses on the binary classification, where each document is assigned to a class or its complement $\mathcal{C} = \{c, \bar{c}\}$.

The important research issue for text classification is how to significantly improve the effectiveness of classifiers in order to handle the large amount of noisy data and address the need for scalability to deal with large-scale text data collections. With noisy text, the correspondence between the feature and class is fuzzy [178]. This problem becomes more severe in deciding relevant and non-relevant information. The important problem related to this issue is how to select relevant features to determine a clear decision boundary between relevant and non-relevant information. The process of text feature selection contains some uncertainties;

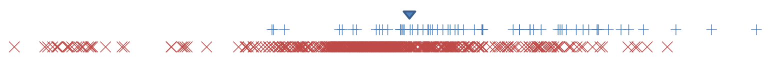


Figure 1.1: Decision boundary in a binary classification.

therefore, most feature selection methods use a weighting function to describe the importance of features. These weights are significant for ranking relevant information; however, so far, an effective way to integrate these weight functions with the existing classifiers has not been found.

Figure 1.1 shows an example of a real binary class dataset topic “Ferry boat sinkings” classified by the Rocchio classifier with inverted triangle markers identifying the optimal decision boundary. In this figure, a plus represents a relevant document and a cross represents a non-relevant document. The number of non-relevant documents is typically much higher than the number of relevant documents. As can be seen in the figure, most of the mixed relevant and non-relevant documents are around the decision boundary. In these documents, a word such as “ferry” appears but not for sinking ferry, or word “sink” appears but not in relation to ferry.

Due to the presence of noisy terms in text documents, the identification of useful features for classification purposes is a challenging issue. Data mining techniques have recently been used for text feature selection, in which the relevance feature discovery (RFD) model [99] has demonstrated excellent performance. One of the interesting findings in relation to RFD is that the best set of features include both specific and general terms; however, most general terms are used in both relevant and non-relevant information, and this leads to an unclear decision boundary between the relevant and non-relevant information. RFD largely reduces noisy terms and achieves excellent performance for ranking documents. However, the use of RFD features to produce a binary classification by setting a decision boundary is not an easy problem.

The text classification process can be conducted by scoring/ranking and decision boundary setting. Decision boundary or threshold setting is often considered as a trivial process and thus is under-investigated. Yiming Yang [175] suggested that the threshold setting is important for text classification, and an effective threshold setting strategy can significantly improve the effectiveness of classification.

A two-stage decision model for information filtering was introduced by Li et al. [103]. In this research, a decision boundary is used in the first stage to solve the mismatch problem. The model used the second stage to solve the overload problem. The problem is that a single threshold can only be used to solve the mismatch problem, but it cannot be used to solve the overload problem.

1.2 Research Questions

After features are selected from a training set, document representations are produced. In this thesis, a document is represented by a weight (or score). Training documents are then ranked based on their scores. After that, a decision boundary is set. In some cases, positive and negative training documents are clearly separated; however, in most cases there are regions in which positive and negative documents are mixed, and on those cases decision boundary is uncertain. With an uncertain boundary, the classification problem is more complex.

In order to solve this issue, this research addresses the following questions:

Research Question 1: How can a model be developed to describe the decision boundary, especially the uncertain decision boundary, and produce an effective binary text classification from an existing feature selection model?

To address this question, a boundary region for each topic is explored. Then, with the information about this region (especially the fences of the region), the decision boundary is calculated. The initial decision boundary is set and then adjusted based on the region's fences. This approach makes minimal usage of experimental parameters. Furthermore, in the case of uncertain boundary, this boundary region is used to identify which new incoming documents should be swapped in the decision based on the specific and generic document vectors.

Reseach Question 2: How can proposed classifier is combined with current other classifiers to be boost classification effectiveness?

After the decision boundary has been set to generate an effective classifier, a further investigation is needed to increase the classification effectiveness. A potential alternative to address this issue is the combination of the proposed text classifier with a current classifier. In the classifier combination, current lower performance classifiers can be used.

This thesis proposes a novel boundary setting method to solve this challenging issue to produce an effective text classifier called RFD$_\tau$. The RFD$_\tau$ model views the incoming document into three regions (namely, low score, boundary and high score regions) rather than two classes (relevant and non-relevant). It also uses an uncertain decision boundary (two thresholds) rather than a clear decision boundary (one threshold) to identify the lower boundary and upper boundary. The RFD$_\tau$ model then groups the features into three categories and represents a document in three vectors to change better decisions for documents in an uncertain decision boundary. This thesis also presents a classifier combination to boost the effectiveness of RFD$_\tau$, using a recall-oriented classifier combined with RFD$_\tau$.

In order to evaluate the proposed model, substantial experiments are conducted on a popular text classification corpus based on the Reuters Corpus Volume 1 (RCV1). The performance of RFD$_\tau$ is compared with the performance of nine types of classifier including state of the art classifiers. The results show that the proposed model outperforms the baseline classifiers.

1.3 Contributions and Significance

The main contribution of this thesis is the development of an effective model that deal with the uncertain decision boundary for text classification. The proposed decision boundary model uses only training set with minimal experimental parameters, which makes it efficient. Using existing pattern-based feature selection RFD, the proposed decision boundary setting produces the RFD$_\tau$ classifier. Even the initial decision boundary setting version developed in this study with no experimental parameters produces better performance than baseline models.

Another contribution of this thesis is the proposition of a two stage approach to combine two existing classifiers. This combination is used to increase the performance of the proposed RFD$_\tau$ classifier.

This research produced an effective text classifier. Text classification is an important task in text mining. With the abundance of text in real world, this research has significant contribution.

The main evaluation criterion in this thesis is classifier effectiveness, compared to popular and state of the art classifiers. The conducted experiments show that the proposed RFD$_\tau$ classifier outperforms baseline classifiers.

In proposed decision boundary setting, clear and uncertain boundary are identified. In clear boundary, the minimum score of the training relevant document is higher than the maximum score of training non-relevant document; otherwise the boundary is uncertain. With different actions for clear and uncertain boundary, decision boundary setting is more effective. In proposed classifier combination, an effective classifier was produced by combine recall oriented and precision oriented classifiers.

1.4 Publications

Based on the work conducted in this thesis, the following publications have been produced:

- Moch Arif Bijaksana, Yuefeng Li, and Abdulmohsen Algarni. Scoring thresholding pattern based text classifier. In *Proceeding of the 5th Asian Conference on Intelligent Information and Database Systems (ACIIDS 2013)*, Springer Lecture Notes in Computer Science, Berlin, Germany, pages 206-215, 2013.
- Moch Arif Bijaksana, Yuefeng Li, and Abdulmohsen Algarni. A pattern based two-stage text classifier. In *Proceeding of the 9th International Conference on Machine Learning and Data Mining (MLDM 2013)*, Springer Lecture Notes in Computer Science, Berlin, Germany, pages 169-182, 2013.
- Moch Arif Bijaksana, Yuefeng Li, Laurianne Sitbon. A Decision Boundary Setting for Text Classifier. To be submitted to Decision Support System journal.
- Yuefeng Li, Abdulmohsen Algarni, Yan Shen, Mubarak Albathan and Moch Arif Bijaksana. Relevance Feature Discovery for Text Mining, 2014 online published, DOI: http://dx.doi.org/10.1109/TKDE.2014.2373357.

1.5 Thesis Outline

The remainder of this thesis is organised as follows:

Chapter 2 provides a comprehensive review of related works on text classification.

Chapter 3 introduces current pattern-based feature selection model used and its implementation to classification.

Chapter 4 explains the main concept of the proposed decision boundary setting model. This chapter describes how an effective decision boundary for text classification is set.

Chapter 5 presents a technique to increase classification performance by combining classifiers. The proposed classifier model is combined with an existing classifier to produce higher performance.

Chapter 6 presents benchmark dataset, performance measures, baseline models setting, and experiment results. A detailed discussion of the result of experiment is also presented.

Chapter 7 concludes the thesis by summarising important points and findings, and suggests directions for future work.

Chapter 2

Literature Review

This literature review covers four topics that are relevant to the present research: (1) binary classification; (2) document representation; (3) classifier and classification models; and (4) decision boundary setting.

2.1 Binary Classification

Text classification or text categorisation (TC) involves the automatic labelling of text using predefined labels or categories automatically based on a model. The model is constructed from labelled examples of text in a similar problem domain. More formally, text categorisation is a task of assigning a Boolean value to each pair $\langle d_j, c_i \rangle \in \mathcal{D} \times \mathcal{C}$ where $\mathcal{D}$ is a domain of documents and $\mathcal{C} = \{c_1, \ldots, c_{\mathcal{C}}\}$ is a set of predefined classes/categories [51, 144].

In this thesis, we concentrate on binary classification where each document $d_j \in D$ must be assigned either to category c_i or to its complement, $\bar{c}_i$. Theoretically, binary classification is a general form of classification. Multi-class and multi-label problems can be solved by using binary classifications [144].

The SVM and AdaBoost algorithms were originally designed for binary classification [155]. Most artificial neural network classifiers are best suited to learning binary function [41]. Theoretical studies of learning have focused almost entirely on learning binary functions [115, 166].

Many real world datasets, including text are multi-class and multi-label. The most popular text corpora for classification are multi-class, and many of them are also multi-label.

There are many ways to reduce a (single-label) multi-class problem to a binary problem. The most popular and simple ways are comparing each class to the rest (one-against-rest), comparing the classes (one-against-one) [65], and using error correcting codes (ECOC) [41, 56, 155]. After being processed in binary, the result must be combined [155]. Allwein et al. [3] presented comprehensive information on this method, and proposed a unifying approach.

In the one-against-rest approach, a binary dataset for classification is created for each class. In this dataset, all instances that belong to that class are considered to be positive (or relevant) examples, while the remaining instances are considered to be negative (or non relevant) examples. In the one-against-one approach, a binary classification is created for each class with a pair of classes (i.e. that class and another class). Each classifier is used to distinguish between that pair of classes. To get the final decision, a voting scheme is typically employed to combine the predictions, where the class that receives the highest number of votes is assigned to the test instance [155]. A problem appears when the voting result is tied. To solve this problem, a probability value is generated for each decision [155, 159].

For example, to illustrate the one-against-rest and one-against-one approaches, we use a toy multi-class dataset with 10 documents and three classes (see Figure 2.1). This table can be transformed into the one-against-rest approach (see Figure 2.2) and the one-against-rest approach (see Figure 2.3). A binary classification

Document	Class
d_1	c_1
d_2	c_3
d_3	c_2
d_4	c_1
d_5	c_1
d_6	c_2
d_7	c_3
d_8	c_1
d_9	c_2
d_{10}	c_1

Figure 2.1: Original table for multi-class example.

process is applied for each transformed binary dataset.

ECOC employs a distributed output code, which was pioneered by Sejnowski and Rosenberg [145]. In ECOC, each class is assigned a unique binary string of length n referred to as the “codewords” [41]. Then n binary classification is used to predict each bit of codeword string. The final result is defined by the closest Hamming distance of codewords produced by the binary classifiers [41]. An important issue in ECOC is how to design an optimal codeword. An additional advantage of using ECOC is that it can provide reliable class probability estimates [41].

For example, consider a three-class problem with classes c_1, c_2 and c_3. Suppose we encode the classes using a four-bit codeword as illustrated in Figure 2.4. For the multi-class problem in Figure 2.1 and the codeword in Figure 2.4, four datasets can be built for each bit of codeword (Figure 2.5). If a test instance is classified as (1,1,1,1) by these binary classifiers, then this will be predicted as c_3 because its Hamming distance is lowest. The Hammimg distances of that test instance and c_1, c_2 and c_3 are 2, 3 and 1 respectively.

Many text classification applications are binary, such as information filtering [8] and email spam filtering [32]. Basic spam email filtering, it has two classes: spam and no-spam.

Document	Class (c_1)
d_1	Pos
d_2	Neg
d_3	Neg
d_4	Pos
d_5	Pos
d_6	Neg
d_7	Neg
d_8	Pos
d_9	Neg
d_{10}	Pos

(a)

Document	Class (c_2)
d_1	Neg
d_2	Neg
d_3	Pos
d_4	Neg
d_5	Neg
d_6	Pos
d_7	Neg
d_8	Neg
d_9	Neg
d_{10}	Neg

(b)

Document	Class (c_3)
d_1	Neg
d_2	Pos
d_3	Neg
d_4	Neg
d_5	Neg
d_6	Neg
d_7	Pos
d_8	Neg
d_9	Neg
d_{10}	Neg

(c)

Figure 2.2: One-against-rest approach.

Document	Class ($c_1 - c_2$)
d_1	c_1
d_3	c_2
d_4	c_1
d_5	c_1
d_6	c_2
d_8	c_1
d_9	c_2
d_{10}	c_1

(a)

Document	Class ($c_1 - c_3$)
d_1	c_1
d_2	c_3
d_4	c_1
d_5	c_1
d_7	c_3
d_8	c_1
d_{10}	c_1

(b)

Document	Class ($c_2 - c_3$)
d_2	c_3
d_3	c_2
d_6	c_2
d_7	c_3
d_9	c_2

(c)

Figure 2.3: One-against-one approach.

Class	Codeword			
	f_1	f_2	f_3	f_4
c_1	1	0	0	1
c_2	0	0	1	0
c_3	1	1	1	0

Figure 2.4: A four-bit error correcting output code for a three-class problem.

Document	Class (f_1)
d_1	1
d_2	1
d_3	0
d_4	1
d_5	1
d_6	0
d_7	1
d_8	1
d_9	0
d_{10}	1

(a)

Document	Class (f_2)
d_1	0
d_2	1
d_3	0
d_4	0
d_5	0
d_6	0
d_7	1
d_8	0
d_9	0
d_{10}	0

(b)

Document	Class (f_3)
d_1	0
d_2	1
d_3	1
d_4	0
d_5	0
d_6	1
d_7	1
d_8	0
d_9	1
d_{10}	0

(c)

Document	Class (f_4)
d_1	1
d_2	0
d_3	0
d_4	1
d_5	1
d_6	0
d_7	0
d_8	1
d_9	0
d_{10}	1

(d)

Figure 2.5: Example of binary dataset for ECOC.

Document	Class
d_1	c_1
d_2	c_1, c_3
d_3	c_2, c_3
d_4	c_1, c_3
d_5	c_1, c_2
d_6	c_2
d_7	c_3
d_8	c_1, c_2, c_3
d_9	c_2, c_3
d_{10}	c_1, c_3

Figure 2.6: Original table for multi-class example.

For a multi-class dataset with many classes, such as in RCV1 data collection [96] with 103 topic classes, transforming the multi-class into a binary class faces the problem of imbalanced class distribution, where c_i is much smaller than $\bar{c}_i$. A popular solution for the imbalanced class problem is sampling [74].

For the multi-label problem, some transformations to the binary problem can be employed [27, 163, 164]. At least four approaches to transform a multi-label dataset into a single-label dataset have been presented [27], namely, the all label assignment (ALA) approach, no label assignment (NLA) approach, largest label assignment (LLA) approach, and smallest label assignment (SLA) approach. The ALA (also referred to as the binary relevance approach) is a popular transformation method and is usually considered to be the best [27, 164]. It is similar to the one-against-rest approach for multi-class transformation. Classifiers implementing binary relevance for multi-label datasets include ML-kNN [185] and SVM [57].

Figure 2.6 shows a simple multi-label dataset. Binary relevance transformation creates three binary datasets, as illustrated in Figure 2.7.

Document	Class (c_1)
d_1	Pos
d_2	Pos
d_3	Neg
d_4	Pos
d_5	Pos
d_6	Neg
d_7	Neg
d_8	Pos
d_9	Neg
d_{10}	Pos

(a)

Document	Class (c_2)
d_1	Neg
d_2	Neg
d_3	Pos
d_4	Neg
d_5	Pos
d_6	Pos
d_7	Neg
d_8	Pos
d_9	Pos
d_{10}	Neg

(b)

Document	Class (c_3)
d_1	Neg
d_2	Pos
d_3	Pos
d_4	Pos
d_5	Neg
d_6	Neg
d_7	Pos
d_8	Pos
d_9	Pos
d_{10}	Pos

(c)

Figure 2.7: Binary relevance transformation for multi label dataset.

2.2 Document Representation

Natural language text is semi-structured data that cannot be directly used as input for a learning process. A text document has to be transformed into structured data, usually as a set of independent feature values. To illustrate how a document is represented in a feature space, suppose feature set *F* is extracted from a document $d, d \xrightarrow{F} \{f_1, f_2, \ldots, f_{|F|}\}$. Each document d_j is represented by a feature vector as $\vec{d}_j = (w_{1,j}, w_{2,j}, w_{3,j}, .., w_{n,j})$, where $w_{i,j}$ is the weight of the feature. The weight reflects the relative importance of the features.

The features can be simple structures (words), complex linguistic structures (e.g., phrases, lexical dependencies, part of speech), statistical structures (e.g. n-gram, patterns (termsets)) or properties of the document (e.g., document' length).

The set of features consists of one or more types. Most systems use only one kind of feature (e.g., term); however, many works have found that more than one types of feature can increase classification performance [111, 117, 143].

A document representation should cover as much information as possible from the document. On the other hand, it must be suitable as the input representation for a learning algorithm.

2.2.1 Term Features

Term [1] is the most common type of feature in document representation. A complex natural language document is transformed into a set of simple independent terms. Using the simple term feature makes the classification efficient. However, the relational information among the terms is lost [150].

A topic might have clues (good indicators) to represent the topic. These clue terms can be seen as keywords. A keyword has more weight than other terms. The number of clues can be few or many. For example, in the TREC-11 RCV1 corpus, the topic "Economic espionage" (e.g., "spy", "espionage", "industry") has a lower number of good indicators than the topic "Progress in treatment of schizophrenia" (a lot of treatment' jargon). The result of our experiment showed that most classifiers have much higher classification effectiveness for the topic "Economic espionage" than the topic "Progress in treatment of schizophrenia".

In topics with a large number of clues, the term-based approach might not be able to capture the theme of the document, so the classification effectiveness is low [142]. As a solution to this problem, the term co-occurrence approach can be used. Schütze et al. [142] utilised latent semantic indexing.

Another problem when using terms as features is the semantic ambiguity. This can be a polysemy, whereby terms can be used to express different things in dif-

[1]Terms are normalised words. Word normalisation handles superficial differences, such as accents and diacritics, capitalization/case-folding, and other issues in a language (e.g., color vs. colour in English) [105].

ferent contexts (e.g., driving a car and driving results). This type of ambiguity affects precision. It is also manifested as a synonymy, whereby terms can be used to express the same thing (e.g., espionage and spying), which will affect recall.

A more complex statistical form is the n-gram (of terms or characters). Most n-gram based categorisation methods are less efficient, and their effectiveness is not better than term-based methods [25, 144]. However, in a more recent experiment, Ifrim et al. [71] proposed an effective new method with variable-length n-grams that consider both word-level n-grams to capture phrases and character-level n-grams to capture morphological variations (stemming, transcription from non-Latin alphabets, misspelling, etc.).

2.2.2 Natural Language Knowledge Usage

Stavrianou et al. [154] discussed natural language particularity issues for comprehensive text mining. There are at least two ways in which natural language knowledge participates in document representation: directly as features, and indirectly as references in the feature weighting process. When natural language knowledge participates directly as features, the features created are complex linguistic features [111]. Complex linguistic features include document Lemmas, that is, base forms of morphological categories, like nouns (e.g., *bank* from *banks*) or verbs (e.g., *work* from *worked*,*working*); phrases (sentence fragments as word sequences); and word senses (i.e. different meanings of content words, as defined in dictionaries). When natural language knowledge participates in document representation indirectly as references in the feature weighting process, the natural language knowledge, such as the role of semantic in [149], is used to create a structure which is then used in the word feature weighting process. Here, each sentence is labelled by a semantic role labeller to first identify the semantic role in that sentence. The weight of term is calculated based on the term's role in that sentence.

Complex linguistic features can be used as the only feature in document representation or as an addition to existing word features. Most existing works use complex linguistic features together with word features in document representation, rather than purely complex linguistic features, because it is more effective for categorisation [94, 111, 117].

2.2.3 Phrase-Based Representation

In grammatical terms, a phrase is defined as "a group of words which is part rather than the whole of a sentence" [169], such as "take away" and "pull out". Phrases have been used intensively in information retrieval (IR); however, at least in early works, it was not effective [94]. Lewis [95] explained that the phrase is not a good feature because it does not fit the four criteria of a good feature: (i) a small number of indexing terms, (ii) flat distribution of values for indexing terms, (iii) lack of redundancy among terms, and (iv) low noise in indexing term values. However, like other "failed" natural language knowledge forms, there are no detailed quantitative analysis examining why phrases do not successfully improve categorisation effectiveness compared to single word features. Other works [111, 143] explained the failure of phrases in document representation in text classification; however, those analyses are short and descriptive. Regarding criteria (i), Lewis [94] wrote that he used automatic syntactical phrase identification in his experiment and found 32,521 phrases, while only 22,791 words were found. More recent works concluded that syntactical phrases improved precision and recall [52], and helped in generating high-precision classification in a large data collection [6].

Word sense is used to overcome the synonym, polysemy and homonym problems, which are related to word meanings (senses). For example, in case of polysemy, the word “phone” can be a noun referring to a device and a verb meaning to communicate. In English, a popular lexical database which provides the senses of English words is WordNet [108]. Kehagias et al. [82] used a WordNet-based annotated (by linguists) corpus to compare word-based and sense-based features for categorisation. Using a small training set (182 documents), they found that sense-based features did not improve effectiveness significantly. Moschitti [110] concluded that the word-sense feature was not sufficient to improve text categorisation effectiveness. However, another investigation indicated that WordNet’s Synsets relationship hierarchy usage helped categorisation performance [123].

2.2.4 Word-Clustering

The word clustering method is a feature construction approach; it creates a new, reduced-size feature set by joining similar words into clusters. Instead of the words clusters or a representative of them may be used as features in document representation. Some of the earliest works on word clustering for text categorisation were conducted by Lewis [94, 95] and concluded that traditional word clustering was unlikely to contribute to a significant improvement in text categorization.

The distributional clustering of words scheme was introduced by Pereira et al. [124]. Based on an information-theoretic approach, words are represented as distributions over the document class where they appear. An early implementation of distributional-word clustering for feature selection in text categorisation was [5], which used Naive Bayes as the learning algorithm. Slonim and Tishby [151] used the agglomerative approach of information bottleneck method for clustering words, along with Naive Bayes. The information bottleneck method was proposed by Tishby et al. [161]. To improve the performance, Bekkerman et al. [7] used SVM, instead of Naive Bayes.

2.2.5 Latent Semantic Indexing

The latent semantic indexing (LSI) method was developed by Deerwester et al. [38] based on latent semantic analysis (LSA) [43]. In that early work, Deerwester et al. used LSI for document similarity analysis. Briefly, LSI is an implementation for indexing (initially in IR, then in text mining, word sense disambiguation etc).

LSI uses a linear algebra's matrix factorisation, called singular value decomposition, to transform original high-dimensional data to a new lower, orthogonal dimension approximation by applying truncated singular value decomposition to the word-document matrix. This new space is a more compact document representation. Words and documents that are closely associated will be placed near one another in a new "semantic space". LSI can also be seen as soft clustering [105].

These various representations can be higher for document collection produced by several different people. The new set of vectors can be viewed as pseudo document vectors. However the created features are not intuitively interpretable.

If there are a lot of different terms which all contribute to specific information, then it is harder for a term-based classifier to perform with high effectiveness [142]. In natural language text, the user can express a given object using various terms, where a common and obvious phenomenom is the word synonym. For example, in a group of documents discussing cars, besides "cars", we may use "automobile", "auto" or "vehicle". In the bag of words model, each of these words will be separated features. Theoretically, synonymity will affect recall [38], especially in IR, because a document cannot be relevant to a query which has different terminologies even when it has the same meaning.

However, if there are a small number single terms providing a have strong "clue" for the class label, then the term-based feature can easily obtain high effectiveness. For example, in the TREC topic 133 about Hubble Space Telescope, a single word "Hubble" is a good indicator for prediction [142]. On the other hand, LSI may group this key word with other words which makes prediction harder. The availability of such clue terms is higher on a low frequency class.

Another natural language challenge is polysemy, where a word has multiple meanings such as the word "book" that has many different meanings. It can be refer to "text" (noun) as in the sentence "I borrowed this book from university library" and "arrange" (verb) as in "He booked us tickets to see the performance". Many researchers have stated that LSI helps to minimise the synonym and polysemy affects; however, in fact it does not work well for polysemy [38]. Synonym in LSI is a loose meaning for the term co-occurrence.

The LSI features could be additional features on top of other term-based or background knowledge [182].

Another probabilistic model is the latent dirichlet allocation model [15]. The basic idea of this model is that documents are represented as random mixtures over latent topics, where each topic is characterised by a distribution over words [15].

2.2.6 Pattern-Based Document Representation

As term suffers from the problem of synonymy and polysemy, some works use phrases (concatenation of two or more words which must occurs in text separated only by white space) to represent documents. However, phrase-based representation don't yield significant performance improvement. Phrases have large numbers of redundant and noisy phrases among extracted phrases in the documents [143, 144]. Another drawback of phrase is language dependency.

A new approach to document representation is using a set of terms. A set of term in data mining is usually called a termset or pattern. A pattern can be seen as a statistical phrase[144]. Pattern mining is a popular type of data mining [63]. Pattern mining has been extensively studied in data mining communities for many years. Many efficient algorithms has been proposed.

A pattern-based document representation is pattern taxonomy model (PTM) [99, 171, 188]. PTM uses the intra-document-based frequent closed sequential pattern with the paragraph as the working unit (in data mining it is usually called a transactional unit). A pattern is called a frequent pattern if its frequency is greater than a user-specified threshold. A pattern is closed if none of its immediate supersets have exactly the same support count. The pattern taxonomy model defines closed patterns as meaningful patterns because most of the sub-sequence patterns of closed patterns have the same frequency, which means they always occur together in a document. Smaller patterns in the taxonomy, are usually more general because they have a high occurrence frequency in both positive and negative documents; but larger patterns are usually more specific since they have a small chance of being found in both positive and negative documents [99]. The pattern taxonomy model prunes non-closed patterns from document representation in an attempt to reduce the size of the feature set by removing noisy patterns.

2.2.7 Feature Selection and Weighting

Two related tasks in document representation are feature selection and feature weighting. The more information, the more accurate a learning system; however, in the real world, some information can be useless information (e.g., noisy, uninformative, redundant information). In text categorisation, a high-dimension (large feature set) might affect categorisation performance and, reduce effectiveness because of over-fitting and decreases efficiency because of complex computation.

Many term weighting methods in text mining are derived from IR, such as term frequency (tf) and inverse document frequency (idf) [135] and from theoretical and statistics based term weighting methods [37], such as information gain, mutual information, and chi-square. A lot of work has been done on term weighting. There are many current works on weighting methods, including relevance frequency method [88] which is a supervised inter-document method exploiting the distribution of relevant documents in the collection, and the distributional feature method [173], which is an intra document method for words.

In the relevance frequency method, an effective term weighting function is simply calculated based on the number of documents in the positive category that contain this term and the number of documents in the negative category that contain this term [88]. The distributional features method includes the compactness of the appearances and the position of the first appearance of the word. The compactness measures whether or not the appearances of a word concentrate in a specific part or are spread throughout the document. A less compact word has more weight, because it is more likely to be related to the document's topic [173]. The second consideration in the distributional feature method is the position of the first appearance of the word. In a news article this is characterised as an inverted pyramid structure [119]. Therefore, as Xue and Zhou [173] stated if a word is mentioned earlier, it will be more important than other words that are mentioned later.

2.3 Classification Model

2.3.1 Probabilistic Based Classifiers

Probabilistic classifiers use a modelling of probabilistic relationship among features. The probability of a document to its class is computed by the Bayes theorem. The Bayes theorem is a statistical principle of combining the prior knowledge of classes with new evidence from data [144, 155].

In the Bayes theorem, the conditional probability of class c_i for a document d_j is:

$$P(c_i \mid d_j) = \frac{P(c_i) \times P(d_j \mid c_i)}{P(d_j)}$$

As stated in the discussion on section 2.2, a document is represented by a vector of binary or weighted terms $\vec{d}_j = (w_{1,j}, w_{2,j}, w_{3,j}, .., w_{n,j})$

2.3.2 Naive Bayes

The Naive Bayes is a popular Bayesian classifier, with it has a long history as a core technique in information retrieval [93]. Naive Bayes classifiers have been investigated by many authors including Calders and Verwer [24], Langley et al. [91].

The Naive Bayes classifier simplifies learning by assuming that features are independent, and that independence is generally a poor assumption. In spite of the naive simplified assumptions, Naive Bayes classifiers work quite well in many complex real-world situations [26, 78, 130], including text classification [155]. Naive Bayes is often comparable in classification effectiveness with other classifiers [42]. It is an efficient and scalable approach [129] Naive Bayes is very popular in binary classification problems of anti-spam e-mail filters [106].

2.3.2.1 Bayes Network

The Bayesian belief network, also referred to as the Bayes Network or Bayes Net provides a flexible approach by allowing users to set some of the pairs of features to be directed acyclic dependent [68, 155]. The Bayes Net at least has four advantages [67]: it can handle situations where some data entries are missing; it can be used to learn causal relationships; it is an ideal representation for combining prior knowledge and data; and it is an efficient approach for avoiding the overfitting of data [67].

2.3.3 Support Vector Machines

The support vector machines (SVM) has its roots in Vapnikk's [167, 168] statistical learning theory. SVM works well in many areas including in data with high dimensionality such as text [76, 144]. The basic idea of the SVM is to find a decision boundary between two classes that is maximally far from any point in the training data (maximal margin hyperplane) [23, 33, 105]. It represents the decision boundary using a small subset of training data, known as support vectors. Many articles on SVM have been published, including works by Bennett and Campbell [10], Burges [23], Hearst et al. [66], Schölkopf and Smola [141], Tsochantaridis et al. [162].

Joachims [76, 77] introduced SVM method for text classification. Followed by others, including Dumais and Chen [44], Dumais et al. [45], Klinkenberg and Joachims [84].

The decision function in the SVM is defined as:

$$h(x) = sign(w \cdot x + b) = \begin{cases} +1 & \text{if } (w \cdot x + b) > 0 \\ -1 & \text{otherwise} \end{cases}$$

where x is the input object. The training phase of the SVM involves estimating the parameters w and b of the decision boundary from the training data. $b \, \epsilon \, \Re^2$ is a threshold and $w = \sum_{i=1}^{l} y_i \alpha_i x_i$ for the given training data: $(x_i, y_i), ..., (x_l, y_l)$, where $x_i \, \varepsilon \, \Re^n$ and $y_i = +1(-1)$, if document x_i is labelled positive (negative).

$^2\Re$ represents real number

$\alpha_i \,\epsilon\, \Re$ is the weight of the sample x_i and satisfies the constraint:

$$\forall_i : \alpha_i \geq 0 \quad \text{and} \quad \sum_{i=1}^{l} \alpha_i y_i = 0$$

2.3.3.1 Sequential Minimal Optimization

Sequential minimal optimization (SMO) is an algorithm for training an SVM. SMO exploits datasets which contain a substantial number of zero elements. SMO works particularly well for sparse data sets, with either binary or non-binary input data [125].

2.3.4 Decision Tree-Based Classifiers

A decision tree text classifier is a tree in which internal nodes are labelled by features, and leaves are labelled by categories [144]. Overviews of decision tree classifiers can be found in the articles by Breslow and Aha [20], Buntine [22], Moret [109], Murthy [114], Rokach and Maimon [132], Safavian and Landgrebe [134].

Examples of some popular decison tree classifiers include CART [19], ID3 [127], C4.5 [128], and CHAID [81].

For example, Figure 2.8 illustrates a decision tree built by J4.8, a variant of C4.5, for a dataset topic number 102 used in this thesis. This topic contains information pertaining to crimes committed by people who have been previously convicted and later released or paroled from prison.

The main process in the training phase is tree growing (building). In building a decision tree classifier, a decision tree classifier chooses one feature at each node of the tree that most effectively splits into two or more subsets. In this tree growing process internal nodes are split using a splitting function. ID3 and C4.5

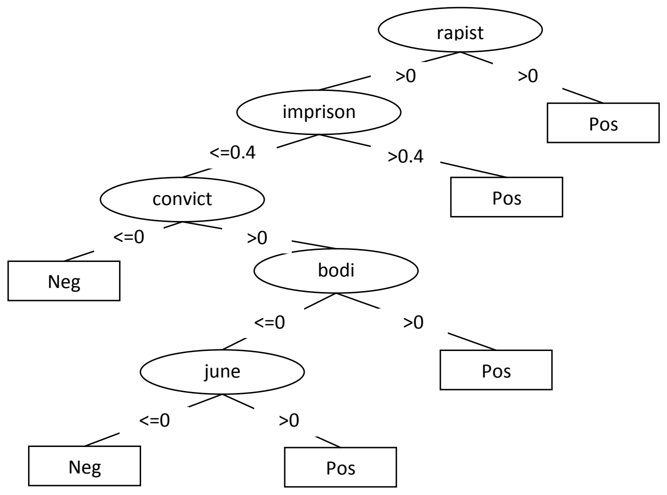


Figure 2.8: A decision tree produced by J4.8 for topic 102.

use entropy, CART uses the Gini index, and CHAID uses statistical χ^2 for the splitting function.

Most decision trees have two branches in node splitting (binary tree) [144]. Most trees split one attribute at a time, in a top-down approach [155]. Landeweerd et al. [90], Pattipati and Alexandridis [120] investigated the bottom up approach.

A common problem in decision tree classifiers is an incorrect generalisation called overfitting [105]; in this case, the problem is classifier overfit (perfectly fit or tuned) for the training set [144]. This problem usually occurs in a complex tree. In a complex tree, overfitting also learns from noise in the training set [105].

A common solution to reduce overfitting is tree pruning [137]. Tree pruning removes the overly specific branches [47, 126]. Therefore, the training phase can involve two sequence processes, namely, tree growing followed by pruning [144].

2.3.5 Decision Rule Based Classifiers

Decision rules are a collection of “IF...THEN..." rules. The left-hand side is the rule antecedent or precondition, and the right-hand side is the rule consequent

IF *rapist* > 0 THEN Pos
ELSE IF *imprison* <= 0.401 AND *lejeun* <= 0.286 AND *earli* <= 0.442 AND *convict* <= 0 THEN Neg
ELSE IF *bodi* > 0 THEN Pos
ELSE IF *june* <= 0 THEN Neg
ELSE Pos

(a) PART

IF *rapist* <= 0 AND *june* <= 0 THEN Neg
ELSE IF *convict* <= 0 AND *kill* <= 0.349 THEN Neg
ELSE Pos

(b) RIPPER

Figure 2.9: Decision rule sets produced by (a) PART and (b) RIPPER for topic 102.

which contains the predicted class. The antecedent takes the conditional disjunctive normal form (DNS) $if < DNFformula > then < category >$.

Decision rules can be generated from decision tree-based classifiers. However, rule-based classifiers tends to generate more compact rules than decision tree classifiers [144].

After a rule set has been produced, a pruning phase to reduce overfitting is applied, where the ability to correctly classify all the training examples is traded for more generality [144].

Some rule-based classifiers have been introduced, for example 1R [69], IREP [53], CN2 [28], and PART [48]. 1R is a simple form of rule-based classifier with only a single rule. However, 1R performs just as well as other classifiers in many datasets. PART builds a partial C4.5 decision tree in each iteration and makes the "best" leaf into a rule. Some decision rule-based classifiers have been applied to text classification, including CHARADE [112], DL-ESC [97], RIPPER [29–31], and SCAR [113].

For example, Figure 2.9 illustrates decision rule sets built by PART and RIPPER for a dataset topic number 102 used in this thesis.

2.3.6 Representative Based Classifiers

One type of classification model is class representation, where there are two representations in each class. The format of the class representation in principle is the same as that of document representation. Among the classifiers discussed in this literature review, there are two classifiers that have classification models in class representation, namely, the Rocchio [72] and pattern-based PTM [171]. Rocchio classification is an adaptation of Rocchio's formula for relevance feedback in information retrieval that was pioneered by Hull [70].

In Rocchio, the class representation is a centroid. The Rocchio algorithm [131] has been widely adopted in the areas of text categorisation. It can be used to build the profile for representing the concept of a topic which consists of a set of relevant (positive) and irrelevant (negative) documents. The centroid $\vec{c}$ of a topic can be generated by using

$$\vec{c} = \alpha \frac{1}{|D^+|} \sum_{\vec{d} \in D^+} \frac{\vec{d}}{||\vec{d}||} - \beta \frac{1}{|D^-|} \sum_{\vec{d} \in D^-} \frac{\vec{d}}{||\vec{d}||}$$

where α and β are empirical parameters, $\alpha + \beta = 1$; and $\vec{d}$ is a document vector.

A centroid is the centre of mass of all the documents in that class. Therefore, the Rocchio classifier is also called a centroid-based classifier [61, 157] or cluster-based classifier [73].

To predict a new document, the Rocchio classifier calculates the cosine similarity, or the Euclidean distance of the normalised document vector between the new document and the class centroids. With such a simple process, the Rocchio classifier is very efficient, during both the training and prediction process. However, Sebastiani [144] showed that the Rocchio classifier is less accurate for text.

According to Manning et al. [105] and Yang [174], the Rocchio is inaccurate in datasets with classes that are not approximately spheres with similar radii (or multiple clusters).

Several studies have shown that Rocchio effectiveness can be improved with some adjustments, such as utilising near positive training documents [138] or with a normalized summed centroid calculation [158].

By producing class representatives, classifiers have obvious advantages in terms of interpretability, as such representatives are more readily understandable by a human [144].

2.3.7 Neural Networks-Based Classifiers

Artificial neural networks or neural networks (NNs) are inspired by biological brain neural systems. As in the brain system, an NN is composed of an interconnected assembly of nodes and directed links. The simplest model of an NN is a linear classifier called a perceptron [36] which has only input and output nodes.

A more complex NN is the nonlinear multi-layer perceptron (MLP) which has one or more additional layers [87, 118, 133]. With hidden layers, the MLP can accommodate a more complex relationship between the input and output variables that the network is able to learn.

In the text domain, the input units represent terms, while the output units represent the category or categories of interest, and the weights on the edges connecting the units represent the dependence relations[144].

For example, Figure 2.10 illustrates an architecture of MLP for a dataset topic number 102 used in this thesis with 10 terms.

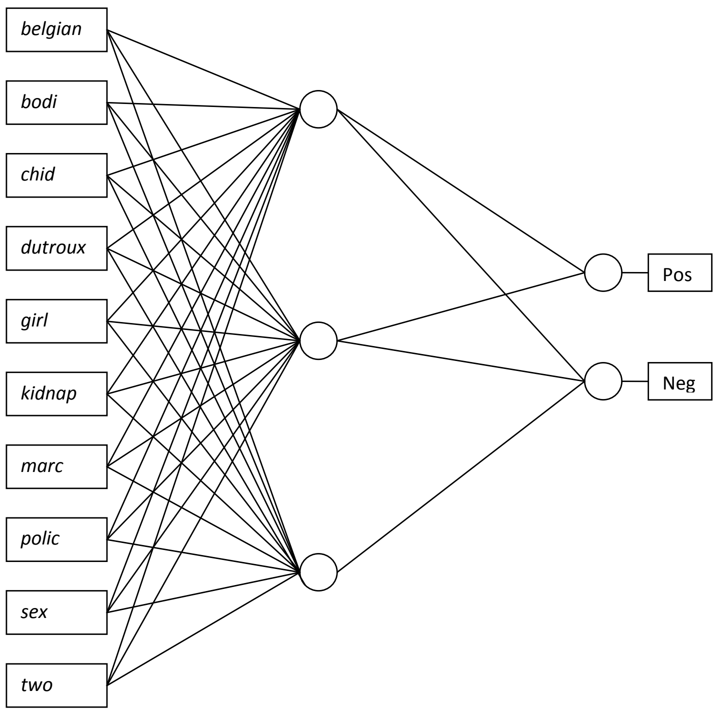


Figure 2.10: MLP architecture for topic 102 with 10 terms.

2.3.8 Instance-Based Classifiers

Most classifiers construct an explicit classification model from the data in the training phase (eager learners). The instance-based classifier delays the classification model construction from the training data until it is needed to classify new instances (lazy learner) [4, 155]. It means that the lazy classifier does not maintain a classification model.

A popular instance-based approach is the k nearest neighbours (kNN) technique, which uses k number of nearest neighbours (k training instances) to identify a new test document to decide the class for a new instance. Therefore, the kNN requires a proximity measure to determine the similarity or distance between two instances. To identify the nearest neighbours, the classifier ranks the training set, and finds the k most similar (k neighbours). A popular similarity function in document is cosine similarity.

Creecy et al. [34] pioneered the application of instance-based classifiers in the text domain [144], followed by others, including Soucy and Mineau [153], Tan [156], Yang and Liu [177], and Aha et al. [1].

An important issue in kNN is choosing the right value of k. Overfitting can occur because of noise(for too small k) and misclassification can occur because of similar training data (for too large k)[155]. Other issues realted to kNN are its inefficiency at classification time [144], sensitivity to the choice of the algorithm's similarity function [1], and the finding nearest neighbours efficiently. [80].

2.3.9 Classifier Combination

A classifier combination is a combination (or ensemble) of multiple base classifiers. It is called also a meta classifier. In a classifier combination, individual decisions are combined in some way (typically by weighted or unweighted voting) to classify new instances [40].

There are two necessary conditions for a classifier combination to perform better than a single classifier: the base classifiers should be as uncorrelated/independent of each other as possible [85, 86, 165], and the base classifiers should do better than a classifier that performs random guessing [155]. In other words, the base classifiers must be accurate and diverse [39, 64]. Two classifiers are diverse if they make different errors on new data points [40].

A popular survey of the classifier combination is in [40], while a survey of commonly used ensemble-based classification techniques is in [79]. An annual conference has been held in the area of classifier combinations since 2000 [3].

Classifier combinations are constructed at least four ways [155]: by manip-

[3]Conference proceedings can be found in
`http://www.informatik.uni-trier.de/ Ley/db/conf/mcs/index.html`

ulating the training set (e.g., with boosting such as AdaBoost), by manipulating the input features (e.g., Random Forest), by manipulating the class labels, and by manipulating the learning algorithms.

The most popular classifier combination methods are bagging and boosting. Bootstrap aggregating (bagging) was proposed in [17]. Bagging is a simple ensemble-based algorithms; however, it has good performance [17]. The same base learning methods (weak learner) is used with different variations. A diversity of classifiers in bagging is obtained by different subsets of the training set, which are randomly drawn with replacements from the training dataset. The final result is obtained by voting with equal weight.

Similar to bagging, boosting [139, 180] also creates an ensemble of classifiers by resampling the data, which are then combined by majority voting. However, in boosting, the same learning methods (weak learners) are sequentially trained, and the training instances that previous models misclassified are emphasised. An example of a popular boosting method is AdaBoost [50].

Random Forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. After a large number of trees is generated, they vote for the most popular class [18].

The classifier combination has been applied to text classification; for example, by Al-Kofahi et al. [2], Bell et al. [9], Bennett et al. [11], Bi et al. [13], Larkey and Croft [92], Li and Jain [101], Yang et al. [179], and Schapire and Singer [140].

2.3.10 Rough Set

Rough set theory dealing with uncertainty and imprecision, it was developed by Pawlak [121] in early eighties. Rough set is an important mathematical tool for managing uncertainty. Information about rough set theory in data mining can be found in [190]. Rough set theory can be used for classification to discover structural relationships within imprecise or noisy data [62].

Rough Set-based has been used for data mining, such as information filtering by Li and Zhong [98], genetic algorithms by Kim [83], rule based classifier by [104, 191]. Sarkar [136] introduced fuzzy-rough uncertainty to enhance classification performance of the kNN algorithm. Pawlak [122] used rough-set theory in the Bayes theorem classifier. Miao et al. [107] proposed a hybrid algorithm based on rough set to combine kNN and the Rocchio to produce a better classification performance.

Some application of rough set based classifiers for text were introduced, include for email spam filtering [187] and news classification [184]. In some classifiers, rough set is used to reduce the number of attribute [75, 100].

2.4 Decision boundary setting

The classification process can be performed in two ways: fully automated classification ("hard" classification) and semi-automated (ranking) classification [144]. In an automated classification process, new incoming instances are automatically labelled by the classifier. In a semi-automated classification process, new incoming instances are ranked. In a semi-automated classification process, classifiers usually produce a score value for each instance. For a text classification with dataset $\mathcal{D}$, the function $Score_i : \mathcal{D} \rightarrow weight(\mathcal{D})$. For example, given a new document d_j, the classifier returns a *score* for it. Documents are then ranked according to the $Score_i$ value [144].

For some automated classifiers like Naive Bayes and Rocchio, the classification process can be viewed as producing score. In Naive Bayes, $Score_i(d_j)$ is defined in terms of a probability; whereas in Rocchio, $Score_i(d_j)$ is a measure of the vector closeness to centroids [144, 175]. These classifiers then perform the decision boundary (threshold) setting, for which default decision boundary value is usually used (e.g., 0.5 for Bayes classifier [58].

For automated classification, the decision boundary setting is often considered a trivial process and is not important; therefore, it has tended to be under-investigated [175]. However, Yang [175] proved that decision boundary setting is important and not simple. Using kNN, Yang [175] proved that an effective decision boundary setting strategy produces significantly better performance than other decision boundary setting strategies.

Existing works on decision boundary setting strategies are generally in the context of the post-processing automatic classification of or for multi-label classification problems, such as [46, 58, 175]. However, in principle, these decision boundary setting strategies can be used to transform ranking (semi-automatic) classification to “hard” (automatic) classification.

If a classifier based on the value of scores can result in effective classification in the expected measurement (e.g., F measure), there is no need to calculate the threshold experimentally [144] (e.g., probability in Bayes classifier [144, 175]). However, according to [175] (based on [12, 55, 176]), the probability values generated by the Bayes classifier are not fully reliable. The Bayes probability value tends toward one or zero exponentially if the number of features in the representation of the documents increase.

Another approach to transforming the classifier score into a class prediction is by transforming the classifier score into a probability estimate [12, 172, 181]; however this is not an easy method [175].

The question that arises is how to calculate the decision boundary experimentally in the training process in order to maximise effectiveness. Yang [174, 175] identified three decision boundary setting strategies, namely, ranking-based, score-based, and proportional-based strategies. Yang [174, 175] referred to them as RCut (rank-based thresholding), SCut (score-based local optimisation), and PCut (proportion-based assignment), respectively. Ranking-based decision boundary setting is referred as fixed decision boundary setting by Sebastiani [144] and as “k-per-doc" decision boundary setting by Lewis [94]. Score-based decision boundary setting is referred to as CSV decision boundary setting by Sebastiani [144] and Schapire et al. [138].

Ranking-based decision boundary setting is used for category-based ranking [4] [144, 175]. With the validation set (separate from the training set and test set), the k largest category score is selected for each selected document, leading to maximum evaluation. There will be the same number of category (k) for each document.

In score-based decision boundary setting, the decision boundary values are local per category. In the validation set, with descending order based on the score, the classifier looks for a decision boundary value in order to get the maximum performance for that category.

Proportional-based decision boundary setting, uses the test set, so this decision boundary setting strategy cannot be applied to online classification (where the new documents appear one by one). In this strategy, it is assumed that the proportions among the categories in the test and training sets are the same. The scores are sorted in the test set. For each category c_i, top-k: $k_j = P(c_i) \times x \times m$ documents are labelled c_i, where $P(c_i)$ is the prior probability of c_i (calculated from the training set), x is a real value for the decision boundary, and m is the number of categories. Then, the decision boundary value is set at x so that the performance on the validation set reaches a global optimum for the classifier.

To improve SVM performance, Shanahan and Roma [147, 148] and Zhai et al.

[4] There are two types of ranking: category ranking and document ranking. The work in this thesis is related to document-ranking.

[183] adjusted the SVM' decision boundary based on utility models and the ranking properties of classifiers using the beta-gamma decision boundary setting technique. The beta-gamma decision boundary setting algorithm relaxes the SVM decision boundary from zero and translates the SVM hyperplane towards the denser class (i.e., the class with more training data).

Genkin et al. [54] presented a simple Bayesian logistic regression approach that uses a Laplace prior to avoid overfitting and produces sparse predictive models for text data, with a novel decision boundary setting named tuned decision boundary setting. In this decision boundary setting technique, decision boundary for each category is set to get the highest training performance. Tuned decision boundary setting outperforms default decision boundary setting for linear regression [160], ridge logistic regression [186] and SVM for text classification.

I many informaytion filtering problems, with only access to the positive training dataset, Li et al. [102] and Zhou et al. [189] proved that the decision boundary can be calculated theoretically based on the training set. In this case, the decision boundary is mainly used to filter the negative documents that are not similar to a positive document. It is assumed that the negative document characteristics are not similar or close to the common features of the topic (positive documents). The similar negative document identification process is also performed on the semi-supervise learning process [51]. Using the rough sets approach to calculate the score of the document, Li and Zhong [98] found that the decision boundary is the minimum weight of the positive documents (all the document in the positive region have a weight at least the same as the minimum weight of the positive documents). Li et al. [102] and Zhou et al. [189] concluded that the topic common features/theme can be obtained from the average weight of positive documents in the training set. Therefore, the decision boundary is the mean of total document weight in positive training documents. However, in a real situation it might be skew of the documents' weight distribution. So, the decision boundary value is $threshold = \bar{m} + \gamma(\alpha + skew)$, where α is the standard deviation and γ is the experimental parameter.

2.5 Summary

In this chapter, we have reviewed several topics related to our project, namely, binary classification, document representation (feature weighting, selection, feature type, dimensionality reduction), classifiers and decision boundary setting.

Binary classification is a popular way to solve multi-class problems. Many studies have used this method.

The selection of document representation affects text classification performance. How a complex text document is transformed into a simple representation is an important and challenging question. A large number of studies on feature weighting and selection can be found. The pattern-based approach is a promising document representation alternative especially for fine tuning. There are many challenging work to optimise the usage of patterns for text classification. Pattern-based feature improves the effectiveness of binary classifiers by identifying useful features, especially when noisy information in text classification is present.

For classification, most research studies have used existing machine learning classifiers. Many of them proved that SVM is the best performer. However, several others classifiers also have also been shown to be competitive.

Decision boundary setting is an important and difficult work in the classification process. The review of literature conducted for the purposes of the present study found that there few studies on decision boundary setting. The existing works show that optimal decision boundary setting can improve classification effectiveness for automated classification. Only a few studies have been conducted on decision boundary setting in order to make a ranked classification method produce decision labels on incoming documents. Most existing existing boundary setting strategies are based on validation and test set. With a validation set, the size of the training dataset will be decreased; this makes training process suboptimal, especially if the the dataset contains a small number of training set. When using a testing set, classifying new documents cannot be done online. In an existing training set based boundary setting strategy, the decision boundary had to be set for every category This motivate us to calculate decision boundaries based on training set for all existing categories.

Several classifier combination models have been proposed, an important issue is how to choose base weak classifiers.

Chapter 3

Pattern-based Feature Selection and Its Application to Classification

Pattern-based feature selection has been developed as an effective scheme in [99, 102, 171]. In this pattern-based feature selection, it uses sequential closed patterns.

3.1 Pattern

For a given topic, the objective of relevance feature discovery in text documents is to find a set of useful features, including patterns (termsets), terms and their weights, in a training set D, which consists of a set of relevant (positive) documents, D^+, and a set of irrelevant (negative)documents, D^-. A document d has a set of paragraphs $PS(d)$.

Let $T_1 = \{t_1, t_2, \dots, t_m\}$ be a set of terms which are extracted from D^+. Given a *termset* X, a set of terms, in document d, $coverset(X)$ is used to denote the covering set of X for d, which includes all paragraphs $dp \in PS(d)$ such that $X \subseteq dp$, i.e., $coverset(X) = \{dp | dp \in PS(d), X \subseteq dp\}$. Its *absolute support*

Table 3.1: Pattern based document representation.

Doc	Patterns
d_1	$\langle carbon \rangle_4, \langle carbon, emiss \rangle_2, \langle air, pollut \rangle_2$
d_2	$\langle greenhous, global \rangle_3, \langle emiss, global \rangle_2$
d_3	$\langle greenhous \rangle_2, \langle global, emiss \rangle_2$
d_4	$\langle carbon \rangle_3, \langle air \rangle_3, \langle air, antarct \rangle_2$
d_5	$\langle emiss, global, pollut \rangle_2$

is the number of occurrences of X in $PS(d)$, that is $sup_a(X) = |coverset(X)|$. Its *relative support* is the fraction of the paragraphs that contain the pattern, that is, $sup_r(X) = \frac{|coverset(X)|}{|PS(d)|}$. A termset X is called a *frequent pattern* if its sup_a (or sup_r) $\geq min\_sup$, a minimum support.

Given a termset X, its covering set $coverset(X)$ is a subset of paragraphs. Similarly, given a set of paragraphs $Y \subseteq PS(d)$, it can be defined its *termset*, which satisfies

$$termset(Y) = \{t | \forall dp \in Y \Rightarrow t \in dp\}.$$

The closure of X is defined as follows:

$$Cls(X) = termset(coverset(X)).$$

A pattern X (also a termset) is called *closed* if and only if $X = Cls(X)$.

Let X be a closed pattern, so

$$sup_a(X_1) < sup_a(X) \tag{3.1}$$

for all patterns $X_1 \supset X$.

These definitions can also be found in [99, 102, 171].

Table 3.1 illustrates document representation in pattern-based model. In this table d_1 has three pattern features $\langle carbon \rangle_4$, $\langle carbon, emiss \rangle_3$, and $\langle air, pollut \rangle_2$.

Subscripted values are support values which represents weight. It means that in d_1 there are four paragraphs contain pattern $\langle carbon \rangle$, three paragraphs contain pattern $\langle carbon, emiss \rangle$, and two paragraphs contain pattern $\langle air, pollut \rangle$.

3.2 Deploying higher level patterns on low-level terms

For term-based approaches, weighting the usefulness of a given term is based on its appearance in documents. However, for pattern-based approaches, weighting the usefulness of a given term is based on its appearance in discovered patterns.

To improve the efficiency of the pattern taxonomy mining, an algorithm, *SP-Mining*(D^+, min_sup) [170], was proposed (also used in [102, 171]) to find closed sequential patterns for all documents $\in D^+$, which used the well-known *Apriori* property in order to reduce the searching space. For all relevant documents $d_i \in D^+$, the *SPMining* algorithm can discover all closed sequential patterns, SP_i, based on a given min_sup.

Let SP_1, SP_2, ..., $SP_{|D^+|}$ be the sets of discovered closed sequential patterns for all documents $d_i \in D^+ (i = 1, \cdots, n)$, where $n = |D^+|$. For a given term t, its deploying support, called $weight$, the discovered patterns can be described as follows [102, 171]):

$$weight_1(t, D^+) = \sum_{i=1}^{n} sup_i(t) = \sum_{i=1}^{n} \frac{|\{p | p \in SP_i, t \in p\}|}{\sum_{p \in SP_i} |p|} \tag{3.2}$$

where $|p|$ is the number of terms in p.

3.3 RFD Model

RFD model [99] for relevance feature discovery describes the relevant features in relation to three groups, namely: positive specific terms, general terms and negative specific terms based on their appearances in a training set.

3.3.1 Specificity function

In the RDF model, a term's specificity (referred to as relative specificity) was defined [99] according to its appearance in a given training set. Let T_2 be a set of terms which are extracted from D^- and $T = T_1 \cup T_2$. Given a term $t \in T$, its $coverage^+$ is the set of relevant documents that contain t, and its $coverage^-$ is the set of irrelevant documents that contain t. It is assumed that the terms frequently used in both relevant documents and irrelevant documents are general terms. The terms that are more frequently used in the relevant documents are classified into the positive specific category; the terms that are more frequently used in the irrelevant documents are classified into the negative specific category.

Based on the above analysis, it is defined the *specificity* of a given term t in the training set $D = D^+ \cup D^-$ as follows [99]:

$$spe(t) = \frac{|coverage^+(t)| - |coverage^-(t)|}{n} \tag{3.3}$$

where $coverage^+(t) = \{d \in D^+ | t \in d\}$, $coverage^-(t) = \{d \in D^- | t \in d\}$, and $n = |D^+|$. $spe(t) > 0$ means that term t is used more frequently in relevant documents than in irrelevant documents.

Classification rules for determining its general terms G, positive specific terms T^+, and negative specific terms T^-:

$$G = \{t \in T | \theta_1 \leq spe(t) \leq \theta_2\},$$

$$T^+ = \{t \in T | spe(t) > \theta_2\},\ and$$

$$T^- = \{t \in T | spe(t) < \theta_1\}.$$

where θ_1 and θ_2 are experimental coefficients.

3.3.2 Weighting features

To improve the effectiveness, RFD uses irrelevant documents in the training set in order to remove the noises. Most models can rank documents (see the ranking function in Equation (3.2) using a set of extracted features. If an irrelevant document gets a high rank, the document is called an offender [98]. The offenders are normally defined as the top-K ranked irrelevant documents. The basic hypothesis is that the relevance features should be mainly discovered from the relevant documents. RFD sets $K = \frac{n}{2}$, as half of the number of relevant documents.

The RDF model uses both the terms' supports and the terms' specificities to define the terms' weights as follows:

$$weight_2(t) = \begin{cases} w(t, D^+)(1 + spe(t)) & t \in T^+ \\ d\_sup(t, D^+) & t \in G \\ d\_sup(t, D^+)(1 - |spe(t)|) & t \in T_1 \\ -d\_sup(t, D^-)(1 + |spe(t)|) & \text{otherwise} \end{cases} \tag{3.4}$$

where the *d_sup* function is defined in Equation (2).

A document can be seen as a vector of term weights $\vec{d} = \langle w_{1j}, w_{2j}, \ldots w_{|T|j} \rangle$, where w_{ij} is weight of term t_{ij}in document. For example in Table 3.1, $D^+ = \{d_1, d_2, \ldots, d_5\}$, term *global* (which appears in document $d_2, d_3, \ldots d_5$), has $rank(global, D^+) = \frac{2}{4} + \frac{1}{3} + \frac{1}{3} = \frac{7}{6}$.

3.4 Application to Text Classification

The advantage of pattern-based feature selection RFD is that it can provide an effective document ranking function for information filtering. Document-ranking function has been derived by exploiting the patterns in the pattern taxonomy. The ranking model sorts a set of documents according to their relevance. Pattern-based approach RFD is better at capturing semantic information without natural language processing. RFD provided a solution to the problems of pattern-based methods which are the low-support problem and misinterpretation problem that means the measures used in pattern mining. The main process of RFD consists of two steps: offender selection (a negative relevance feedback), and the revision of the weights of low-level features (terms) based on both their appearances in the higher level features (patterns). An offender can be used to reduce the side effects of noisy features. This pattern-based feature selection approach outperforms term-based models which are widely used approaches. By using negative relevance feedback, the effectiveness of information filtering can be significantly improved.

RFD is an effective ranking function, therefore an effective text classification is potentially supported by RFD. However, it is hard to use an effective document ranking function for effective text classification. This study proposes a text classification model to extend RFD for effective classification. For a given ranking function, after training documents are ranked, a decision boundary is unlikely decided for a clearly binary classification. The decision boundary can be set based on training documents, verification documents or testing documents. With small number of training set, assigning some of training documents for verification set will harm the performance. While using testing set for decision boundary setting makes online classification cannot be performed. Therefore it is a challenge to utilize RFD for an effective text classification by using only training set for decision boundary setting.

Chapter 4

Decision Boundary Setting

To achieve the best performance in SVM, the objective is to create the decision boundary with the maximum margin. This maximum margin is used to minimize the over-fitting problem. In the Rocchio classifier, the decision boundary is determined from the centroids of each class, where each class has a centroid as a class representative. The decision boundary has the equal distance from two adjacent centroids. The Rocchio classifier uses the criterion cosine similarity or Euclidean distance function. The NB classifier uses probability as the score of documents. In NB, with normalised probability (i.e. the total probability of all classes is one), the decision boundary is basically similar to the Rocchio approach. The decision boundary has equal probability deviation from each class.

After the features are selected and weighed by using pattern-based feature selection and weighting, the weighted terms are then used as document representation, as in many other classifiers. Some classifiers such as Naive Bayes (NB), Rocchio and SVM apply a decision boundary to identify incoming documents.

This chapter presents a new effective boundary setting. An overview of the decision boundary setting approach for a classifier is first provided. The notion of the decision boundary region is then presented. Issues regarding how to set and adjust the decision boundary based on the decision boundary region are discussed. Finally, further efforts to improve classification performance are explained.

At least three issues arise in decision boundary setting, namely, the dataset used, the calculation function, and the number of experimental parameters. For the dataset, a training set, validation set or testing set can be used. When using a validation set, the size of the training dataset will be decreased; this makes training process sub-optimal, especially if the dataset contains a small number of training samples. When using a testing set, classifying new documents cannot be done online. Therefore, the ideal choice for decision boundary setting is to use training set, because of both using training set and validation set. For the calculation function, the calculation can be complex (such as finding decision boundary based on the maximum performance of the training set), or it can be simple (such as finding the minimum of document weight). The simpler calculation function can be the more efficient and the more portable. The number of experimental parameters is similar to the calculation function, that is, the fewer number of experimental parameter can be the more efficient and the more portable.

To clearly understand the concept of the structure used in the proposed model, three regions in the training set are defined. A method for using the three regions in the testing set is then discussed.

4.1 Three Regions

Let D be a training set of documents, which consists of a set of relevant documents, D^+ ; and a set of non-relevant documents, D^-; and $T = \{(t_1, w_1), (t_2, w_2), \ldots, (t_m, w_m)\}$ be a set of term-weight pairs produced by the pattern-based feature extraction method in D (see Chapter 3).

Let U be a testing set of documents. The score of a document d in either D or

U can be calculated based on terms in the document as follows:

$$score(d) = \sum_{t_i \in d \cap T} w_{t_i}$$

Both the training set and the testing set can be separated into three regions based on the document scores. Three regions are defined in the training set, D, namely, the *low score region* ($\mathcal{L}$), the *boundary region* ($\mathcal{B}$) and the *high score region* ($\mathcal{H}$). The ranges of these boundaries are defined as follows:

$$\mathcal{D}_{\mathcal{L}} = \{d \in D | score(d) < \tau_{low}\}$$
$$\mathcal{D}_{\mathcal{B}} = \{d \in D | \tau_{low} \leq score(d) \leq \tau_{high}\}$$
$$\mathcal{D}_{\mathcal{H}} = \{d \in D | score(d) > \tau_{high}\}$$

where τ_{low} and τ_{high} are the lower boundary and upper boundary of $\mathcal{B}$ and are calculated based on the scores of the training documents.

4.2 Boundary Region

An effective way to decide the lower and upper boundaries is based on the minimum score of the relevant documents (τ_P) and the maximum score of the non-relevant documents (τ_N).

$$\tau_P = \min_{d \in D^+} \{score(d)\}$$

$$\tau_N = \max_{d \in D^-} \{score(d)\}$$

The values of τ_{low} and τ_{high} are calculated as follows:

$$\tau_{low} = min(\tau_P, \tau_N)$$

$$\tau_{high} = max(\tau_P, \tau_N)$$

Figure 4.1 shows that a good document scoring models produce a trend of classification effectiveness (F_1 and Acc) with a maximum peak. The maximum peak is in a region between the minimum and the maximum of training documents score, that is $\min_{d \in D}\{score(d)\} < score(d) < \max_{d \in D}\{score(d)\}$. With $\min_{d \in D}\{score(d)\} < \tau_{low} < \max_{d \in D}\{score(d)\}$ and $\min_{d \in D}\{score(d)\} < \tau_{high} < \max_{d \in D}\{score(d)\}$, it means that classification effectiveness will be better around the middle of the document weight distribution where the boundary region most probably located.

The examples in Figure 4.1 use artificial data with $|U^{+}| \approx |U^{-}|$. The figure shows six different probability combinations of positive documents and negative documents in a testing set. In the leftmost tables in the figure, the vertical axis (Y coordinates) represents the positive decision probability value of the documents (documents are predicted as positive) on the horizontal axis (X coordinates). For example, the top chart shows the random case, where the probability of all the documents is 0.5. Further analysis for this phenomenon is set out in Chapter 6.

4.2.1 Use of the Three Regions in the Testing Set

Let $U_{\mathcal{L}}$ be the low score region of the testing set U, $U_{\mathcal{H}}$ be the high score region of U, and $U_{\mathcal{B}}$ be the boundary region of U. For incoming documents in U, the simply way is to use the lower and upper boundaries to classify U into three regions:

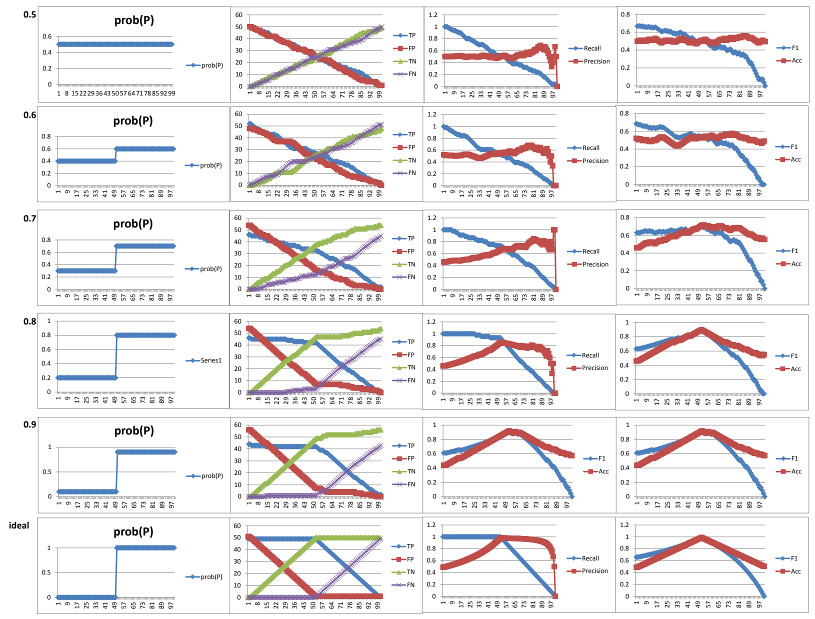


Figure 4.1: Performance in several different decision boundaries.

$$U_{\mathcal{L}} = \{d \in U | score(d) < \tau_{low}\}$$

$$U_{\mathcal{B}} = \{d \in U | \tau_{low} \leq score(d) \leq \tau_{high}\}$$

$$U_{\mathcal{H}} = \{d \in U | score(d) > \tau_{high}\}$$

It means that the decision boundary values calculated in the training set can be applied directly for the testing set.

Usually the size of the testing set U is larger than the size of the training set D. Therefore, the region $U_{\mathcal{B}}$ determined in the above equation is only a subset of the real boundary region. Figure 4.2 shows a possible case for the three regions in both the training set and the testing set.

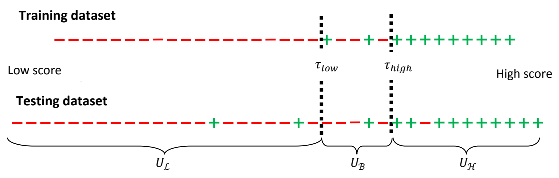


Figure 4.2: Low score, boundary, and high score regions.

4.3 Decision Boundary Setting

After the boundary region has been identified, the next step is finding decision boundary (τ) in and around the near boundary region. First, the initial decision boundary (τ') is selected; then, it is adjusted to improve the classification performance.

4.3.1 Initial Decision Boundary (τ') Setting

To minimise the experimental parameters, the parameters are chosen among the borders of the boundary region, τ_{low}, τ_{high}, τ_P, or τ_N as alternatives for initial decision boundary (τ').

Based on [98], with only the scores of the positive training documents D^+ available, the optimal threshold is τ_P. In a real dataset, in most cases the maximum score of the negative testing document is more than the minimum score of the positive testing document (see Figure 4.3). Therefore, $\tau' < \tau_P$. A simplified version is $\tau_P \leq \tau \leq \tau_N$. With both D^+ and D^- available, the most suitable initial decision boundary is found to be: $\tau' = \tau_{low}$.

4.3.2 Decision Boundary Adjustment

To optimise the performance, the initial decision boundary should be adjusted. With the final decision boundary (τ) in boundary region $\tau_{low} \leq \tau \leq \tau_{high}$, the

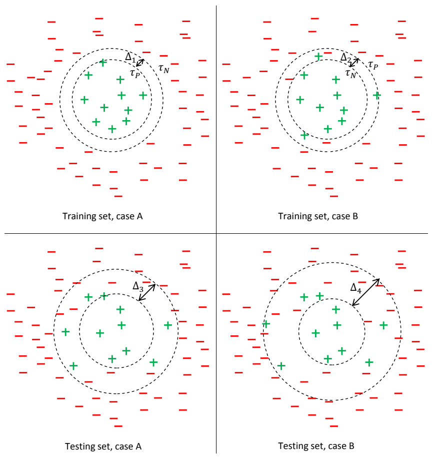


Figure 4.3: Training and testing cases. Case A is a non-overlap training score $\tau_P > \tau_N$, case B is an overlap training $\tau_P < \tau_N$. In both case A and case B testing score are overlap, and usually $\Delta_3 < \Delta_4$.

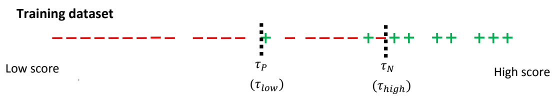


Figure 4.4: Outlier in training set.

adjustment can be calculated based on:

$$\tau = \tau^{'} + (\gamma \times (\tau_{high} - \tau_{low}))$$

where γ is an experimental parameter.

4.3.2.1 Outlier Handling

In the previous section, the initial decision boundary was adjusted directly based on τ_{low} and τ_{high}. To make a better adjustment, some distribution of the weight of the training documents should be considered. This section discusses the influence of outliers in the training dataset on the decision boundary adjustment.

In statistics, an outlier is an observation that lies an abnormal distance from other variables [59]. Figure 4.4 illustrates a positive document that is potentially an outlier. A popular definition of statistical outlier is based on the quartiles of a ranked set of data values. The first/lower quartile (Q1) is defined as the middle number between the smallest number and the median of the data set. The second quartile (Q2) is the median of the data. The third/upper quartile (Q3) is the middle value between the median and the highest value of the data set. The difference between the upper and lower quartiles is called the interquartile range (IQR). The outer fences are Q1 - 3 IQR (lower fence), and Q3 + 3 IQR (upper fence), while the inner fences are Q1 - 1.5 IQR (lower fence), and Q3 + 1.5 IQR (upper fence). All the observations outside the fences are possible outliers; an observation is a suspected/mild outlier if it is outside the inner fence, and an observation is an outlier if it is outside the outer fences [49, 146].

A simple approach to considering outliers in the decision boundary calculation is to remove the outliers. It means that a training document which is considered as an outlier cannot be assigned as τ_{high} or τ_{low}. However, a potential problem with this approach can arise when the number of training documents, especially $|D^{+}|$, is very low.

4.4 Performance Improvement

In the previous section, the training and testing set were divided into three regions, namely, the low score $\mathcal{L}$, high score $\mathcal{H}$, and boundary $\mathcal{B}$ regions. Among these three regions, $\mathcal{B}$ has the highest blended of positive and negative documents. Therefore, the effort to improve classifier perfomance are concentrated on $\mathcal{B}$.

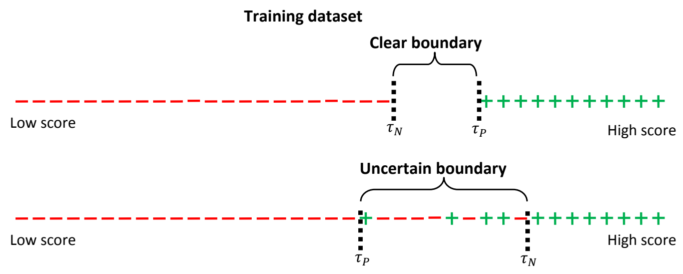


Figure 4.5: Clear and uncertain boundary.

4.4.1 Improving Performance Using Positive, Negative, and General Vectors in Uncertain Boundary

For most real data, documents with different classes cannot be clearly separated. There is a class mixed or interleaved region, especially around the decision boundary. This problem occurs because some documents are outlier or noisy, or because the document representation itself cannot perfectly reflect the semantic meaning of the documents.

To deal with this mixed class issue, instead of using only a *clear boundary* such as in some other classifiers, the proposed method employs an *uncertain boundary* which has a high rate of mixed documents with different classes. The boundaries are determined based on the score of training documents. There are two benefits of using an uncertain boundary. Firstly, the selected features for representing relevant (or non-relevant) information can be clearly understood; secondly, another representation method it can be introduced to further classify the uncertain boundary.

The boundary region can be uncertain where $\tau_P < \tau_N$, or clear where $\tau_P > \tau_N$ (see Figure 4.5). In Figure 4.5 and in the subsequent figures, the positive and negative symbols represent documents. A positive symbol (+) represents a relevant document, and a negative symbol (-) represents a non-relevant document.

Documents are sorted based on their score, descending from the right.

It is obvious that the conditions for a clear boundary are τ_{low} = τ_N and τ_{high} = τ_P. However, for an uncertain boundary, the conditions are τ_{low} = τ_P, and τ_{high} = τ_N. The uncertain and clear boundaries also have the following properties:

Property 1. *For the clear boundary, we have $\mathcal{B} = \varnothing$, $\mathcal{H} = D^+$, and $\mathcal{L} = D^-$.*

Property 2. *For the uncertain boundary, we have $\mathcal{B} \neq \varnothing$, $\mathcal{H} \subseteq D^+$, and $\mathcal{L} \subseteq D^-$.*

In the case of the uncertain boundary, $\mathcal{B} \neq \varnothing$ and $U_{\mathcal{B}}$ is usually larger than $\mathcal{B}$. Therefore, it is impossible to make a clear binary decision even in $\mathcal{B}$ by using the decision boundary τ (see Eq. 4.3.2).

To improve the performance of the classifier, in the uncertain boundary $U_{\mathcal{B}}$ we decompose each document vector into three vectors. As described in Chapter 3, the set of terms T can be grouped into three categories (i.e., T^+, G and T^-) by using the classification rules. Therefore, for a given document vector $\vec{d} = \{(w_{t_1}), (w_{t_2}), \ldots, (w_{t_m})\}$, we can obtain the three vectors, namely, *the positive vector*, *general vector*, and *negative vector*:

$$\begin{aligned} \vec{d}_{T^+} &= \{(w_{t_i}) \in \vec{d} | t_i \in T^+\} \\ \vec{d}_G &= \{(w_{t_i}) \in \vec{d} | t_i \in G\} \\ \vec{d}_{T^-} &= \{(w_{t_i}) \in \vec{d} | t_i \in T^-\} \end{aligned} \tag{4.1}$$

Three scores are then calculated for document d:

$$\begin{aligned} score(\vec{d}_{T^+}) &= \textstyle\sum_{(w_{t_i}) \in \vec{d}_{T^+}, t_i \in d} w, \\ score(\vec{d}_G) &= \textstyle\sum_{(w_{t_i}) \in \vec{d}_G, t_i \in d} w, \\ score(\vec{d}_{T^-}) &= \textstyle\sum_{(w_{t_i}) \in \vec{d}_{T^-}, t_i \in d} w. \end{aligned} \tag{4.2}$$

Finally, the following decision rules for swapping documents $d \in U_{\mathcal{B}}$ are known:

$$(\alpha \times (score(\overrightarrow{d}_{T+}) + score(\overrightarrow{d}_G))) < score(\overrightarrow{d}_{T-}) \Rightarrow d \in U^-$$

$$(score(\overrightarrow{d}_{T+}) + score(\overrightarrow{d}_G)) > (\alpha \times score(\overrightarrow{d}_{T-}) \Rightarrow d \in U^+)$$

where α is an experimental parameter.

4.4.2 Algorithms

The proposed classification model is called RFD$_\tau$. Algorithm 1 and Algorithm 3 describe RFD$_\tau$ in the learning phase and classifying phase, respectively.

Algorithm 1 describes the learning process of the proposed model by using the scoring of the pattern-based feature selection, RFD, and the decision boundary setting based on training documents. For the input of the algorithm, both positive (D^+) and negative (D^-) training documents are required. An experimental parameter, γ, is needed to find the decision boundary. The main outputs of the learning algorithm are the decision boundary value τ, the minimum value of positive training document score τ_P, and the maximum value of negative training document score τ_N. Meanwhile τ_{low} and τ_{high}, are derived from τ_P and τ_N and are used to calculate τ and to improve classification performance in the classifying algorithm. The first steps in the learning phase (steps 1-3) are the score calculations of all the training documents based on the documents' term weights. The weights of the terms are calculated based on patterns. Thereafter, in steps 4 and 5, τ_P and τ_N, and then τ_{low} and τ_{high} are calculated. Finally in step 6, the decision boundary value τ is calculated based on τ_{low},τ_{high}, and experimental parameter γ. The time complexity of this algorithm is $O(|T| \times |d| \times |D|)$.

Algorithm 2 describes learning with outliers removal, so it prevents outliers to be τ_P and τ_N. However if the number of positive or negative training document is small, there will be no outlier removal. Step 5-12 identify outlier for positive

```
Input  : A training set, $D = D^+ \cup D^-$; and
         experimental parameter, $\gamma$.
Output: Decision boundary value, $\tau$;
         minimum score of training positive document, $\tau_P$;
         maximum score of training negative document, $\tau_N$; and
         border values of boundary set, $\tau_{low}$, $\tau_{high}$.
1 forall the $d \in D$ do
2     $score(d) = \sum_{t \in D} weight_2(t)$
3 end
4 $\tau_P = \min_{d_i \in D^+}\{score(d_i)\}$; $\tau_N = \max_{d_i \in D^-}\{score(d_i)\}$ ;
5 $\tau_{low} = min(\tau_P, \tau_N)$; $\tau_{high} = max(\tau_P, \tau_N)$ ;
6 $\tau = \tau_{low} + (\gamma \times (\tau_{high} - \tau_{low}))$;
```

Algorithm 1: RFD$_\tau$ Learning

training documents for $|D^+ > m|$, while step 15-22 identify outlier for negative training documents for $|D^- > n|$. Where m and n are experimental parameters.

Algorithm RFD$_\tau$ Classfiying (Algorithm 3) shows how to apply RFD$_\tau$ model. This algorithm applies decision boundary value calculated in Algorithm 1 for new incoming documents. For the input of algorithm are testing set U, decision boundary value τ and its related values τ_P, τ_N, τ_{low}, τ_{high}, and an experimental parameter γ. The output of this algorithm are sets of positive and negative label of training docs, POS and NEG. In step 1, it starts with assign new empty sets POS and NEG. Then, for all testing set, as stated in step 3, score of documents are calculated. After that in step 4-6, if the document score less than or equal to decision boundary the document is assigned as negative, otherwise positive. Then start from step 8 to 16, is applied if the topic is uncertain topic and document is in boundary region. Steps 9-11 if an incoming document is initially predicted as positive, but has strong characteristics as negative then this document is swapped from positive to negative. The similar case describes in step 12-14 for when an incoming document is initially predicted as negative, but has strong characteristics as positive then this document is swapped from negative to positive.

```
Input  : A training set, $D = D^+ \cup D^-$; and
         experimental parameters, $m, n, \gamma$.
Output: Decision boundary value, $\tau$;
         minimum score of training positive document, $\tau_P$;
         maximum score of training negative document, $\tau_N$; and
         border values of boundary set, $\tau_{low}$, $\tau_{high}$.
1  forall the $d \in D$ do
2      $score(d) = \sum_{t \in D} weight_2(t)$;
3  end
   // Identify outlier for positive training docs.
4  if $D^+ > m$ then
5      let $D^+ = \{d_0^+, d_1^+, \ldots, d_m^+\}$ in ascending ranking order,
6      $Q_1^+ = |D^+| \times 0.25$;
7      $Q_3^+ = |D^+| \times 0.75$;
8      $d_{Q_1}^+ = \{d \mid d \in D^+, rank(d) = \lceil Q_1^+ \rceil\}$;
9      $d_{Q_3}^+ = \{d \mid d \in D^+, rank(d) = \lceil Q_3^+ \rceil\}$;
10     $IQR^+ = score(d_{Q3}^+) - score(d_{Q1}^+)$;
11     $D_{outlier}^+ = \{d_i \mid d \in D^+, score(d_i) < ((score(d_{Q1}^+) - 1.5 \times IQR^+))\}$;
12     $D^+ \leftarrow D^+ - D_{outlier}^+$;
13 end
   // Identify outlier for negative training docs.
14 if $D^- > n$ then
15     let $D^- = \{d_0^-, d_1^-, \ldots, d_n^-\}$ in ascending ranking order,
16     $Q_1^- = |D^+| \times 0.25$;
17     $Q_3^- = |D^+| \times 0.75$;
18     $d_{Q_1}^- = \{d \mid d \in D^-, rank(d) = \lceil Q_1^- \rceil\}$;
19     $d_{Q_3}^- = \{d \mid d \in D^-, rank(d) = \lceil Q_3^- \rceil\}$;
20     $IQR^- = score(d_{Q3}^-) - score(d_{Q1}^-)$;
21     $D_{outlier}^- = \{d_i \mid d \in D^-, score(d_i) > ((score(d_{Q1}^-) + 3.5 \times IQR^+))\}$;
22     $D^- \leftarrow D^- - D_{outlier}^-$;
23 end
24 $\tau_P = \min_{d_i \in D^+}\{score(d_i)\}$; $\tau_N = \max_{d_i \in D^-}\{score(d_i)\}$ ;
25 $\tau_{low} = min(\tau_P, \tau_N)$; $\tau_{high} = max(\tau_P, \tau_N)$ ;
26 $\tau = \tau_{low} + (\gamma \times (\tau_{high} - \tau_{low}))$;
```
Algorithm 2: RFD$_\tau$ Learning with outlier removal

In document swapping it use score of term specific (see Eq.4.2). The time complexity of this algorithm is $O(|T| \times U)$.

```
Input  : New incoming unlabel documents in testing set, $U$;
         decision boundary value, $\tau$;
         experimental parameter $\alpha$;
         minimum score of training positive document, $\tau_P$;
         maximum score of testing negative document, $\tau_N$; and
         border values of boundary set, $\tau_{low}$, $\tau_{high}$.
Output:  Sets of positive and negative label of training docs, POS and
         NEG.
1  NEG = $\emptyset$, POS = $\emptyset$;
2  forall the $d \in U$ do
3      if score($d \leq \tau$) then
4          NEG = NEG $\cup$ $\{d\}$;
5      else
6          POS = POS $\cup$ $\{d\}$;
7      end
8      if $\tau_P < \tau_N$ and $d \in U_{\mathcal{B}}$ then
9          if $score(d) \geq \tau$ and
           $(\alpha \times (score(\overrightarrow{d}_{T+}) + score(\overrightarrow{d}_G)) < score(\overrightarrow{d}_{T-}))$ then
10             POS = POS - $\{d\}$; NEG = NEG $\cup$ $\{d\}$ ;
11         end
12         if $score(d) < \tau$ and
           $((score(\overrightarrow{d}_{T+}) + score(\overrightarrow{d}_G)) > \alpha \times score(\overrightarrow{d}_{T-}))$ then
13             NEG = NEG - $\{d\}$; POS = POS $\cup$ $\{d\}$ ;
14         end
15     end
16 end
```

Algorithm 3: RFD$_\tau$ Classifying

Chapter 5

Boosting Performance using the Classifier Combination

After the decision boundary has been set to produce an effective text classifier, the next challenge is to boost the performance of RFD_{τ}. This chapter illustrates the method of increasing the RFD_{τ} effectiveness by combining it with another existing classifier. A report on initial work of this combination was provided in [14]. In order to fully discuss the boosting method in this chapter, an overview of the classification combination is first provided. The system architecture and algorithms are then presented.

5.1 Classifier Combination

A classifier combination (also referred to as an ensemble, committee or meta-classifier) is a combination of two or more existing classification systems in order to improve effectiveness. The construction of a classifier combination can occur in at least four ways [155]: by manipulating the training set (e.g. boosting such as AdaBoost), by manipulating the input features (e.g. Random Forest), by

Table 5.1: Combination of two classifiers

		Classifier1	
		Pred P	Pred N
Classifier 2	Pred P	Pred P	Pred N or P
	Pred N	Pred P or N	Pred N

Table 5.2: Combination of two classifiers: main and booster classifier.

		Booster classifier	
		Pred P	Pred N
Main classifier	Pred P	Pred P	Pred N or P
	Pred N	Pred P or N	Pred N

manipulating the class labels, and by manipulating the learning algorithms.

The basic idea of a classifier combination is that a problem requiring expert knowledge will be better solved by a committee of experts rather than by an individual expert [144]. For example, a strong boosting classifier [140] can be built from a combination of the same weak classifiers (weak learners). In this thesis, the approach to constructing a classifier combination by manipulating the learning algorithms is taken. A classifier is chosen to increase effectiveness of RFD$_\tau$. This classifier can be said to be a booster classifier for RFD$_\tau$, with RFD$_\tau$ as the main classifier.

In combining of two classifier models, the main concern is what will happen when the classifiers make different decisions. For two binary classifiers, if one classifier predicts a new document as positive and the other classifier predicts a new document as negative, then the final prediction can be positive or negative (see Table 5.1).

In this thesis, the idea of combining of two binary classifiers is used to increase the performance of RFD$_\tau$, as set out in Table 5.1. An existing classifier that is weaker in overall effectiveness (e.g., F$_1$ or accuracy) but has strong par-

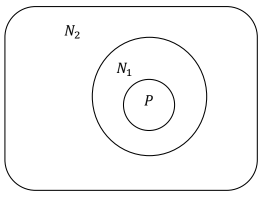


Figure 5.1: Positive P, and negative N_1 (near positive), N_2 in a binary class.

Table 5.3: Classifier combination: recall oriented

		Booster classifier: recall oriented	
		Pred P	Pred N
Main classifier	Pred P	Pred P	Pred N
	Pred N	Pred N	Pred N

tial effectiveness (e.g., recall and precision [1]) is used to boost the effectiveness of RFD$_\tau$ especially in the hard near positive region.

The set of negative documents has a variety of topics. The set of negative documents N is divided into two parts, N_1 and N_2. N_1 (near negative documents) are documents that have close similarity to positive documents P (see Figure 5.1), while N2 are the remaining documents in the set.

In a combination like the one shown in Table 5.2, a conflicting prediction can be solved by using a weighted parameter, or simply by basing on the decision on one base classifier as shown in Table 5.3 and Table 5.4.

The booster classifier can be low score-oriented or high score-oriented (see Figure 5.2). Low score-oriented classifiers concentrate on low score; that means

[1]For F$_1$, accuracy, recall and precision will be presented more detail later in Chapter 6

Table 5.4: Classifier combination: precision oriented

		Booster classifier: precision oriented	
		Pred P	Pred N
Main classifier	Pred P	Pred P	Pred P
	Pred N	Pred P	Pred N

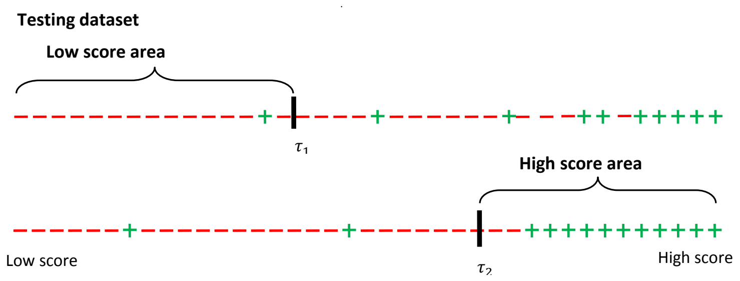


Figure 5.2: Low high areas.

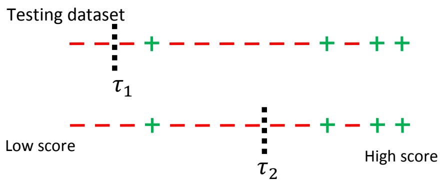


Figure 5.3: Recall oriented

minimising the positive documents in the low score area, but with not too short low score area, FN $\approx$ 0, and low FP (concentrating on the prediction of new documents as negative). Meanwhile, high score-oriented classifiers concentrate on high score; that means minimising negative documents in high score area, but with not too short high score area, FP $\approx$ 0, and low FN (concentrating on the prediction of new documents as positive).

A recall-oriented classifier is a low score-oriented type, and a precision-oriented classifier is a high score oriented type. A recall (or precision) oriented classifier has high recall (or precision) with a moderate precision (or recall). In a classifier combination with recall-oriented booster classifier (see Table 5.3), all new documents that are predicted as negative by booster classifier will be predicted as negative in the final decision of classifier combination. Figure 5.3 illustrates a recall oriented prediction. In the top case, with decision boundary τ_1 high recall but low precision is produced; meanwhile, the case with decision boundary τ_2 produces lower recall and higher precision.

Table 5.5: Classifier combination: detail.

		Booster classifier: recall oriented			
		TP$_{booster}$	FP$_{booster}$	TN$_{booster}$	FN$_{booster}$
Main classifier	TP$_{main}$	TP			FN
	FP$_{main}$		FP	TN	
	TN$_{main}$		TN	TN	
	FN$_{main}$	FN			FN

A more detailed contingency table for a recall-oriented classifier combination is shown in Table 5.5. The goal is to increase final true prediction (TP and TN) and to decrease false prediction (FP and FN).

5.2 System Architecture and Algorithm

Figure 5.4 shows the classification combination framework proposed in this study. By using the same training set, each stage produces a classification model. In the classifying phase, the classification model on classifier one (booster classifier) concentrates on identifying negative documents. At this stage, the documents that are predicted as negative documents are grouped into TN_1 (true negative group one) if the documents are true negative, or grouped into FN_1 (false negative group one) if the documents actually are positive documents. At this stage, the priority is to minimise the FN rate, with acceptable FP (false positive, i.e. negative documents falsely predicted as positive documents) rate. Then, classification model two, which is produced in stage two, is used to identify the documents that were positively predicted in stage one. In the proposed classifier combination model, true negative $TN = TN_1 + TN_2$, false negative $FN = FN_1 + FN_2$, true positive TP, and false positive FP.

The proposed classifier combination (RFD$_{CC}$) uses a recall-oriented Rocchio classifier to boost the RFD$_{\tau}$ classifier. The learning and classifying phases of

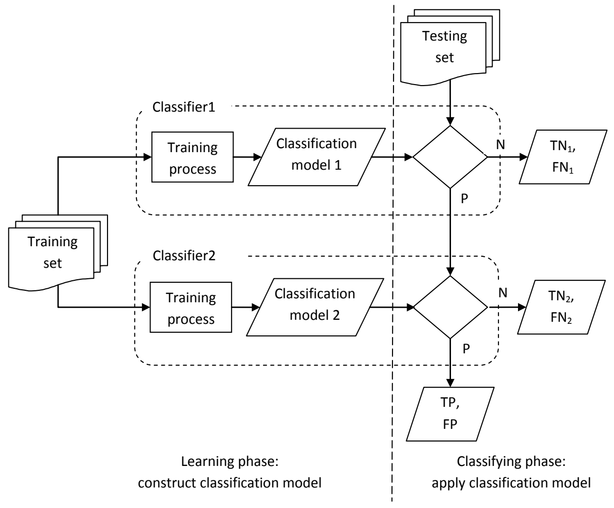


Figure 5.4: Two-stage framework.

Input : A training set, $D = D^+ \cup D^-$.
Output: Rocchio classification model; and threshold value, τ.

```
  // Learn training dataset using Rocchio
     classifier, get Rocchio model.
1 $Model_{Rocchio} = Classifier_{Rocchio}(D)$ ;
  // Calculate the score of training documents using
     $RFD$.
2 $D_{score} = RFD(D, min\_sup, \theta_1, \theta_2)$ ;
  // Calculate the threshold value, $\tau$.
3 $\tau = Thresholding(D_{score})$ ;
```

Algorithm 4: RFD$_{CC}$ Learning

```
  Input  : A new unlabel document;
           Rocchio classification model; and
           threshold value, $\tau$.
  Output: Class label for unlabel document.
  // Predict label the new documents $d_{unlabeled}$ by using
     Rocchio model.
1 $d_{labeled} = Rocchio(d_{unlabeled}, Model_{Rocchio})$ ;
  // If Rocchio label it as negative, so the final
     label of the documen is negative
2 if $d_{labeled}$ is negative then label of $d$ is negative;
  // If Rocchio label it as positive, so the final
     label of the documen is depend on $RFD_\tau$
3 else $d_{labeled} = RFD_\tau(d_{unlabeled}, Model_{Rocchio})$ ;
```
Algorithm 5: RFD$_{CC}$ Classifying

the RFD$_{CC}$ algorithms are outlined in Algorithm 4 and Algorithm 5. For Algorithm 4, in step 1 the algorithm start with build model from Rocchio classifier. Then, generate RFDτ model with generate score for all training documents (step 2) and calculate decision boundary τ (step 3). In Algorithm 5 The first step in classifiying phase (step 1) is class label prediction of the incoming document with Rocchio classifier. If Rocchio classifier predicts the document as negative, then the document is labeled as negative (step 2); otherwise, the label of the document depends on prediction from RFDτ (step 3).

Chapter 6

Evaluation

This chapter addresses the design issues in the experiments for evaluating the proposed models. The experiments conducted to evaluate the proposed classification models and assess the proposed hypotheses are described. The dataset, evaluation metrics and baseline models are described, and the experimental results are presented. At the end of the chapter, the results are analysed and discussed.

The preceding chapters introduced the decision boundary setting and classifier combination models. Two hypotheses have been proposed in this research:

- A decision boundary can be set based on training data for an effective text classification model.
- A classifier combination can be used to improve effectiveness of text classification.

A popular version of the Reuters document collection is chosen from among several versions as our benchmark dataset. Standard performance measures, namely, the F measure and accuracy with macro-averaging and micro-averaging [116, 144], are used to evaluate the experimental performance. Macro-averaging computes a simple average over classes. Micro-averaging pools per-document decisions across classes, and then computes an effectiveness measure based on the pooled contingency table. The experiment results are compared with a comprehensive baseline model. The discussion and analysis of the experiments are presented in two parts based on the models in the previous chapters.

Regarding the first hypothesis, the results led to the following main point: the performance of the proposed classification model (RFD$_\tau$) is significantly better compared to the baseline classification models based on effectiveness. Regarding the second hypothesis, the performance of a classification combination model (Rocchio-RFD$_\tau$) is better than base classification model (RFD$_\tau$) in effectiveness.

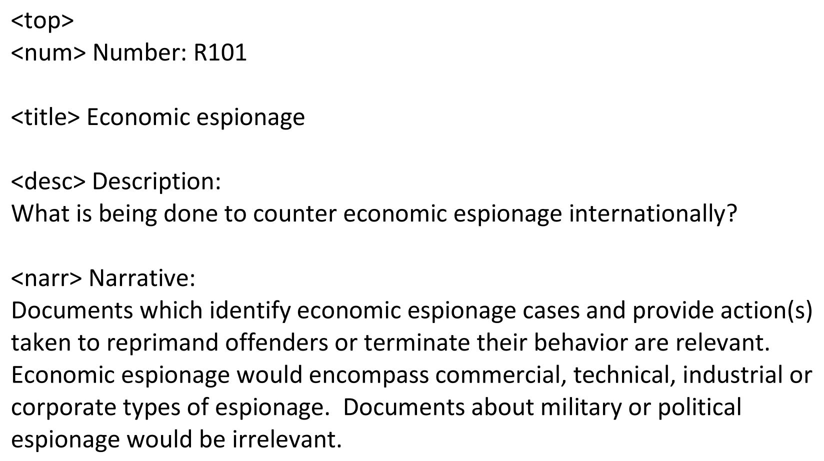
The prototype of proposed models, and two baseline models (Rocchio and Rough Set) are coded in Java programming language. For all other baseline models, Weka software [60] is used. For SVM, the present study used $LibSVM$ package [1] run from Weka. All experiments reported in this thesis were conducted on a PC equipped with an Intel Core2 Duo 3.00GHz,3.21 GB of RAM running a Windows XP operating system.

6.1 Dataset

For text classification, some standard benchmark collections are publicly available for experimental purposes. The most widely used is the Reuters collection, consisting of a set of new articles. The Reuters collection accounts for most of the experimental work in text classification to date [144]. The existing Reuters collections are Reuters-22173, Reuters-21578 [2], and the latest is Reuters Corpus Volume 1 (RCV1) [96]. RCV1 is a collection of English language news articles which were produced by Reuters journalist for the period between 20 August 1996 and 19 August 1997. These documents are formatted using a structured

[1] `http://svmlight.joachims.org/`

[2] `http://www.daviddlewis.com/resources/testcollections/reuters21578/`


<top>
<num> Number: R101

<title> Economic espionage

<desc> Description:
What is being done to counter economic espionage internationally?

<narr> Narrative:
Documents which identify economic espionage cases and provide action(s) taken to reprimand offenders or terminate their behavior are relevant. Economic espionage would encompass commercial, technical, industrial or corporate types of espionage. Documents about military or political espionage would be irrelevant.

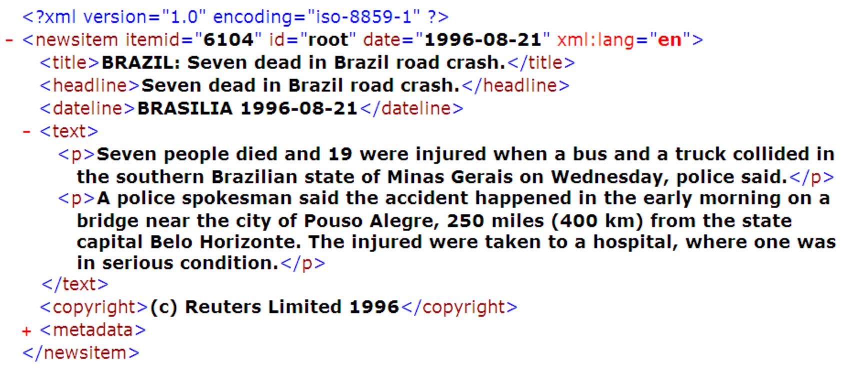
Figure 6.1: Topic statement for the first topic (Topic number 101).

XML scheme.

The present study used TREC-11 Filtering Track RCV1[3], a binary classification version of RCV1. TREC (Text REtrieval Conferene) has developed and provided 100 topics. The first 50 topics were composed by human researchers and the rest were formed by intersecting two Reuters topic categories. The assessor topics typically more reliable than the artificial intersection topics [152]. The 50 assessor topics of dataset contains 21,605 documents, which is a reasonable number of documents for text classification experiment. According to Buckley and Voorhees [21], 50 topics are stable and enough for high quality experiments. Each topic in the dataset is binary class with its own positive and negative set. Each topic has topic statement. Figure 6.1 illustrates a topic statement.

Figure 6.2 shows an RCV1 document. Each document is identified by unique item ID, title and content. The content is divided in paragraphs. The main statistics of dataset are shown in Table 6.1. As shown in the table, the dataset is imbalanced as the number of negative documents is much higher than the number of positive documents. The imbalance rate around 20%.

[3]`http://trec.nist.gov/data/t2002_filtering.html`



```
<?xml version="1.0" encoding="iso-8859-1" ?>
- <newsitem itemid="6104" id="root" date="1996-08-21" xml:lang="en">
    <title>BRAZIL: Seven dead in Brazil road crash.</title>
    <headline>Seven dead in Brazil road crash.</headline>
    <dateline>BRASILIA 1996-08-21</dateline>
  - <text>
      <p>Seven people died and 19 were injured when a bus and a truck collided in
        the southern Brazilian state of Minas Gerais on Wednesday, police said.</p>
      <p>A police spokesman said the accident happened in the early morning on a
        bridge near the city of Pouso Alegre, 250 miles (400 km) from the state
        capital Belo Horizonte. The injured were taken to a hospital, where one was
        in serious condition.</p>
    </text>
    <copyright>(c) Reuters Limited 1996</copyright>
  + <metadata>
  </newsitem>
```

Figure 6.2: An RCV1 XML document.

The documents are treated as plain text documents by preprocessing the documents. The tasks of removing stop-words by reference to a given stop-words list and stemming terms by applying the Porter Stemming algorithm are conducted.

Table 6.1: Statistics of TREC-11 RCV1 dataset.

Topic ID	Training Set			Testing Set		
	$\|D^+\|$	$\|D\|$	$\frac{\|D^+\|}{\|D\|}$	$\|U^+\|$	$\|U\|$	$\frac{\|U^+\|}{\|U\|}$
101	7	23	0.30	307	577	0.53
102	135	199	0.68	159	308	0.52
103	14	64	0.22	61	528	0.12
104	120	194	0.62	94	279	0.34
105	16	37	0.43	50	258	0.19
106	4	44	0.09	31	321	0.10
107	3	61	0.05	37	571	0.06
108	3	53	0.06	15	386	0.04
109	20	40	0.50	74	240	0.31
110	5	91	0.05	31	491	0.06
111	3	52	0.06	15	451	0.03
112	6	57	0.11	20	481	0.04
113	12	68	0.18	70	552	0.13
114	5	25	0.20	62	361	0.17
115	3	46	0.07	63	357	0.18
116	16	46	0.35	87	298	0.29

Continued on next page

Table 6.1 – continued from previous page

Topic ID	Training Set			Testing Set		
	$\lvert D^+\rvert$	$\lvert D\rvert$	$\frac{\lvert D^+\rvert}{\lvert D\rvert}$	$\lvert U^+\rvert$	$\lvert U\rvert$	$\frac{\lvert U^+\rvert}{\lvert U\rvert}$
117	3	13	0.23	32	297	0.11
118	3	32	0.09	14	293	0.05
119	4	26	0.15	40	271	0.15
120	9	54	0.17	158	415	0.38
121	14	81	0.17	84	597	0.14
122	15	70	0.21	51	393	0.13
123	3	51	0.06	17	342	0.05
124	6	33	0.18	33	250	0.13
125	12	36	0.33	132	544	0.24
126	19	29	0.66	172	270	0.64
127	5	32	0.16	42	238	0.18
128	4	51	0.08	33	276	0.12
129	17	72	0.24	57	507	0.11
130	3	24	0.13	16	307	0.05
131	4	31	0.13	74	252	0.29
132	7	103	0.07	22	446	0.05
133	5	47	0.11	28	380	0.07
134	5	31	0.16	67	351	0.19
135	14	29	0.48	337	501	0.67
136	8	46	0.17	67	452	0.15
137	3	50	0.06	9	325	0.03
138	7	98	0.07	44	328	0.13
139	3	21	0.14	17	253	0.07
140	11	59	0.19	67	432	0.16
141	24	56	0.43	82	379	0.22
142	4	28	0.14	24	198	0.12
143	4	52	0.08	23	417	0.06
144	6	50	0.12	55	380	0.14
145	5	95	0.05	27	488	0.06
146	13	32	0.41	111	280	0.40
147	6	62	0.10	34	380	0.09
148	12	33	0.36	228	380	0.60
149	5	26	0.19	57	449	0.13
150	4	51	0.08	54	371	0.15
Total	639	2704		3484	18901	
Max.	135	199	0.68	337	597	0.67

Continued on next page

Table 6.1 – continued from previous page

Topic ID	Training Set			Testing Set		
	$\|D^+\|$	$\|D\|$	$\frac{\|D^+\|}{\|D\|}$	$\|U^+\|$	$\|U\|$	$\frac{\|U^+\|}{\|U\|}$
Min.	3	13	0.05	9	198	0.03
Average	12.8	54.1	0.2	69.7	378	0.19

6.2 Baseline Models and Setting

In order to make a comprehensive evaluation, nine types of classifier for the baseline model were chosen (Table 6.2, Table 6.3), with a total of 22 models (Table 6.4). The proposed model is referred to as the RFD$_\tau$.

6.2.1 Parameter Setting

6.2.1.1 SVM

We used all variants of available kernel types:

Model 1: linear $(u' * v)$

Model 2: polynomial $(\gamma * u' * v + coef_0)^{degree}$

Model 3: radial basis function ($\exp(-\gamma * |u - v|^2)$)

Model 4: sigmoid $(\tanh(\gamma * u' * v + coef_0))$

Other parameters, we used defaults:

- SVM type: C-SVC

[4]We use LibSVM implementation, `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[5]J48 is an open source Java implementation of the C4.5 algorithm [128] in the Weka data mining tool

[6]In Weka implementation, polynomial function in SMO kernel is different with polynomial function in LibSVM

Table 6.2: Type and algorithm of baseline models.

Type	Classifier
Function based	SVM[4], SMO
Classifiers committee based	AdaBoost
Decision tree based	J48[5], Random Forest
Probabilistic based	Naive Bayes, BayesNet
Instance-based (lazy learner)	IBk (KNN)
Neural network based	Multi Layer Perceptron
Decision rule based	PART
Representative based	Rocchio
Information retrieval based	Rough set

Table 6.3: Algorithm of baseline models and their parameters.

Classifier	Parameters
SVM, SMO	Kernels function
AdaBoost	Base classifiers
J48	Tree pruned and unpruned
Naive Bayes	Distribution for numeric attributes
IBk (KNN)	The number of nearest neighbours
Multi Layer Perceptron	The number of hidden layers
Rough set	Threshold setting

Table 6.4: Baseline models.

No	Model	Abbreviation
1	SVM with linear kernel	SVM linear
2	SVM with polynomial kernel	SVM poly
3	SVM with radial basis function kernel	SVM radial
4	SVM with sigmoid kernel	SVM sigmoid
5	SMO with normalized polynomial kernel	SMO norm-poly
6	SMO with polynomial kernel [6]	SMO poly
7	SMO with Puk kernel	SMO Puk
8	AdaBoostM1 with decision stump as its base classifiers	ABM1 base d. stump
9	AdaBoostM1 with decision J48 as its base classifiers	ABM1 base J48
10	J48 with pruned tree option	J48 pruned
11	J48 with unpruned tree option	J48 unpruned
12	Bayesian Network	BayesNet
13	Naive Bayes with normal distribution for numeric attributes	NB normal distr
14	Naive Bayes with kernel density estimator for numeric attributes	NB kernel density
15	Random Forest	Random Forest
16	IBk with the number of nearest neigbours is one	IBk k=1
17	IBk with the number of nearest neigbours is two	IBk k=2
18	Multilayer Perceptron with the number of hidden layer is one	MLP hidden=1
19	Multilayer Perceptron with the number of hidden layer = a where a = (the number of attribs + the number of classes) $/$ 2)	MLP hidden $= a$
20	PART	PART
21	Rocchio	Rocchio
22	Rough Set	RS

- degree in kernel function = 3.
- γ: in kernel function = 1/number of fatures.
- coef_0 in kernel function = 0.
- the parameter C of C-SVC, epsilon-SVR, and nu-SVR (default 1)
- the parameter nu of nu-SVC, one-class SVM, and nu-SVR (default 0.5)
- no normalize input data.
- the epsilon in loss function of epsilon-SVR = 0.1.
- tolerance of termination criterion = 0.001.

6.2.1.2 SMO

We used all variants of kernel types:

Model 1: Normalized Polykernel.

Model 2: Polykernel.

Model 3: Puk.

Other parameters, we used defaults:

- The complexity constant C = 1.
- Normalize training data.
- The tolerance parameter = 1.0e-3.
- The epsilon for round-off error = 1.0e-12.
- The number of folds for the internal cross-validation (use training data).

- The random number seed = 1.
- For kernel function Polykernel
  - The Exponent to use = 1.0.
  - Not using lower-order terms.
- Normalize training data.
- The tolerance parameter =1.0e-3.

6.2.1.3 AdaBoostM1

We used variants of base classifiers: Decision Stump and J48.

Model 1: Base classifier: Decision Stump.

Model 2: Base classifier: J48.

Other parameters, we used defaults:

- Use resampling for boosting.
- Random number seed = 1.
- Number of iterations = 10.

6.2.1.4 J48

We used variants of tree pruning options:

Model 1: Pruned tree.

Model 2: Unpruned tree.

Other parameters, we used defaults:

- Do collapse tree.
- Confidence threshold for pruning = 0.25.
- The minimum number of instances per leaf = 2.
- Don't reduced error pruning.
- The number of folds for reduced error pruning = 3. One fold is used as pruning set.
- Do not use binary splits only.
- Perform subtree raising.
- Clean up after the tree has been built.
- Don't use Laplace smoothing for predicted probabilities.
- Use MDL correction for info gain on numeric attributes.
- The number of seed for random data shuffling: 1.

6.2.1.5 Naive Bayes

We used all default values:

- Estimator algorithm: SimpleEstimator.
- Do not use ADTree data structure.
- Search algorithm: K2.

6.2.1.6 Bayesian Network

We used variants of numeric attributes handling:

Model 1: Use kernel density estimator for numeric attributes.

Model 2: Use normal distribution for numeric attributes.

Other parameter, we used default:

- Use supervised discretization to process numeric attributes.

6.2.1.7 Random Forest

We used all default values:

- Number of trees to build=10.
- Number of features to consider=0.
- Seed for random number generator=1.
- The maximum depth of the trees is unlimited.

6.2.1.8 IBk

We used variants of the number of nearest neighbours:

Model 1: The number of nearest neighbours = 1.

Model 2: The number of nearest neighbours = 2.

Other parameters, we used defaults:

- No distance weighting.
- Use linear search for nearest neighbour search algorithm

6.2.1.9 Multilayer Perceptron

We used variants of the number of hidden layer:

Model 1: The number of hidden layer = 1.

Model 2: The number of hidden layer = a. Where a = the number of attribs + the number of classes.

Other parameters, we used defaults:

- Learning Rate for the backpropagation algorithm = 0.3.
- Momentum Rate for the backpropagation algorithm = 0.2.
- Number of epochs to train through = 500.
- Percentage size of validation set to use to terminate training (if this is non zero it can pre-empt num of epochs = 0.
- The value used to seed the random number generator = 0.
- The consequetive number of errors allowed for validation testing before the netwrok terminates = 0.
- Normalizing a numeric class.
- Normalizing the attributes.
- Learning rate decay will not occur.

6.2.1.10 PART

We used all default values:

- Minimum number of objects per leaf = 2.

- Confidence threshold for pruning = 0.25.
- Seed for random data shuffling = 1.
- Number of folds for reduced error pruning = 3.
- Do not use binary splits only.
- Use MDL correction for info gain on numeric attributes.

6.2.1.11 Rocchio

We used cosine similarity to compare two vectors (documents or centroids).

6.2.1.12 Rough Set

We used variants of threshold setting:

Model 1: min(min(weight(D^+)), max(weight(D^-))).

Model 2: average(weight(D^+)).

Model 3: min(weight(D^+)).

Model 4: max(weight(D^-)).

Model 5: use proportional threshold setting, that is $\frac{U^+}{U} = \frac{D^+}{D}$.

6.3 Feature Weighting and Selection

The feature weighting scheme and selection are important aspects in text classification [89]. Some popular term weighting methods in text mining are derived from information retrieval, such as term frequency (TF) and inverse document frequency (IDF) [135].

In this study, two term weighting schemes were used for the baseline model, namely, traditional term weighting scheme, TF×IDF, and a current text categorisation term weighting relevance frequency (RF) scheme [88]. TF×RF is an effective and efficient term weighting scheme for text classification. It shows a consistently better performance than other term weighting methods [88].

The experiment involved 10, 50, 100, 150, 200 and 250 selected terms, and all terms. In the experiment, the best performance was chosen for each term weighting schema (with priority to the F_1 macro-average for each model).

6.4 Measures

Text classification effectiveness is measured by two different means, namely, F_β and accuracy (Acc), with F_β being the more important metric [144]. F_β is a harmonic mean of recall (R) and precision (P):

$$F_\beta = \frac{(\beta^2 + 1) \times (P \times R)}{\beta^2 \times (P + R)}$$

The parameter $\beta = 1$ was used in the experiment, which means that the recall and precision were weighed equally:

$$F_1 = \frac{2 \times P \times R}{P + R}$$

With the use of the harmonic mean, F places an emphasize on the importance of small values. For example, if the recall is one and the precision near zero, then the arithmetic mean is 0.5, while the harmonic means will be close to zero [35].

To obtain the final result for several topics, two different ways were adopted, namely, micro-averaging (F_1^μ) and macro-averaging (F_1^M) [116, 144] (Table 6.5 and Table 6.6).

Table 6.5: The Contingency table for topic $\mathcal{C}_i$.

Category c_i		Expert judgment	
		Yes	No
Classifier judgment	Yes	TP_i	FP_i
	No	FN_i	TN_i

Table 6.6: The global contingency table.

Category $\mathcal{C} = \{c_i, \ldots, c_{\lvert\mathcal{C}\rvert}\}$		Expert judgment	
		Yes	No
Classifier judgment	Yes	$TP = \sum_{i=1}^{\lvert\mathcal{C}\rvert} TP_i$	$FP = \sum_{i=1}^{\lvert\mathcal{C}\rvert} FP_i$
	No	$FN = \sum_{i=1}^{\lvert\mathcal{C}\rvert} FN_i$	$TN = \sum_{i=1}^{\lvert\mathcal{C}\rvert} TN_i$

The differences between the two methods can be significant. Macro-averaging gives equal weight to each class, whereas micro-averaging gives equal weight to each per-document classification decision [105]:

$$F_1^{\mu} = \frac{2 \times (P^{\mu} \times R^{\mu})}{(P^{\mu} + R^{\mu})}$$

where

$$P^{\mu} = \frac{TP}{TP + FP} = \frac{\sum_{i=1}^{|\mathcal{C}|} TP_i}{\sum_{i=1}^{|\mathcal{C}|}(TP_i + FP_i)}$$

$$R^{\mu} = \frac{TP}{TP + FN} = \frac{\sum_{i=1}^{|\mathcal{C}|} TP_i}{\sum_{i=1}^{|\mathcal{C}|}(TP_i + FN_i)}$$

TP (true positive) refers to the number of documents which the system correctly identifies as positives; TN (true negative) refers to the number of documents which the system correctly identifies as negatives; FP (false positive) refers to the number of documents which the system falsely identifies as positives; FN (false negative) refers to the number of positive documents which the system fails to identify; and $|\mathcal{C}|$ is the number of topics:

$$F_1^{M} = \frac{\sum_{i=1}^{|\mathcal{C}|} F_{1,i}}{|\mathcal{C}|}$$

Table 6.7: Example 1 Macro- and micro-averaging.

MODEL 1							
Topic	TP	FP	TN	FN	Recall	Precision	F_1
1	200	10	50	120	0.625	0.952	0.755
2	3	1	5	1	0.750	0.750	0.750
					0.688	0.851	F_1^M = 0.752
Sum	203	11	55	121	0.627	0.949	F_1^μ = 0.755
MODEL 2							
Topic	TP	FP	TN	FN	Recall	Precision	F_1
1	300	10	50	20	0.938	0.963	0.952
2	1	1	5	3	0.250	0.500	0.333
					0.594	0.734	F_1^M = 0.643
Sum	301	11	55	23	0.929	0.965	F_1^μ = 0.947

where $F_{1,i}$ is the F_1 for topic i.

The accuracy is calculated by the following equations:

$$Acc = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Acc^\mu = \frac{\sum_{i=1}^{|\mathcal{C}|}(TP_i + TN_i)}{\sum_{i=1}^{|\mathcal{C}|}(TP + FP + TN + FN)}$$

$$Acc^M = \frac{\sum_{i=1}^{|\mathcal{C}|} Acc_i}{|\mathcal{C}|}$$

Table 6.7 and Table 6.8 present examples of the differences in the macro-average and micro-average of F_1. These examples use a two-topic dataset. In the first example (Table 6.7), the macro-average of model one outperforms model two; however, the micro-average of model two is better than the micro-average of model one. The second example (Table 6.8) shows two cases. In this example, a difference in the decision only exist in topic one. The result in the first case is $F_1^M < F_1^\mu$, while the result in the second case is $F_1^M > F_1^\mu$. The results of case one and case two are significantly different.

Table 6.8: Example 2 Macro- and micro-averaging.

CASE 1							
Topic	TP	FP	TN	FN	Recall	Precision	F_1
1	200	10	50	20	0.909	0.952	0.930
2	1	1	5	2	0.333	0.500	0.400
					0.621	0.726	$F_1^M = 0.665$
Sum	201	11	55	22	0.901	0.948	$F_1^\mu = 0.924$
CASE 2							
Topic	TP	FP	TN	FN	Recall	Precision	F_1
1	20	10	50	200	0.091	0.667	0.160
2	1	1	5	2	0.333	0.500	0.400
					0.212	0.538	$F_1^M = 0.280$
Sum	21	11	55	202	0.094	0.656	$F_1^\mu = 0.165$

.

The statistical method, the paired two-tailed Student *t-test*, is also used to analyse the experimental results [16]. In statistical hypothesis testing, a probability value (p-value) is used to decide whether there is enough evidence to reject the null hypothesis and whether the research hypothesis is supported by the data. If the associated p-value is low (< 0.05), it shows that the difference in means across the paired observations is significant.

6.5 Evaluation of Decision Boundary Setting

6.5.1 Evaluation Procedures

The proposed model can be applied to the task of text classification to evaluate its effectiveness. The classification process, including evaluation, is illustrated in Figure 6.3. Cleaning (removing single letters that are not meaningful terms), stop-word removal, and stemming are done in the pre-processing stage. In the document representation process, documents are converted to document weights

Table 6.9: Balance vs. imbalance testing set.

Testing set	Predict all P					Predict all N				
	Rec	Prec	F_1	Acc^M	Acc^μ	Rec	Prec	F_1	Acc^M	Acc^μ
Original imbalance	100%	0%	0%	19%	18%	0%	0%	0%	81%	82%
Balance	100%	0%	0%	50%	50%	0%	0%	0%	50%	50%

(scores). This document weight is the representation of a document. The output of the training process is a decision boundary (τ) as its classification model. In the testing process the weight of the testing documents and the decision boundary were compared.

To measure accuracy, under-sampling was used on $|U|$ to make $|U^+| = |U^-|$. Therefore, if $|U^+| < |U^-|$, then U^- was randomly selected, and if $|U^+| > |U^-|$, then U^+ was randomly selected. Five sets of random under-sampling were used for each topic. For accuracy, the use of the original testing dataset with an imbalanced number of positive and negative documents, produced a misleading measurement. An example in the TREC-11 RCV1 dataset is shown in Table 6.9.

Table 6.9 shows if all the testing documents as negative, the recall obtained is 0%, precision is 0%, F_1 is 0%, Acc^M is 81%, and Acc^μ is 82%. If all the testing documents are predicted as positive, the recall obtained is 100%, precision is 0%, F_1 is 0%, Acc^M is 19%, and Acc^μ 18%. Therefore, the average of five random balanced testing was used, where the number of positive documents and the number of negative documents was the same. In the balanced testing set, if all the testing document are predicted as negative, the recall obtained is 0%, precision is 0%, F_1 is 0%, $\text{Acc}^M \approx 50\%$, and $\text{Acc}^\mu \approx 50\%$. If all the testing documents are predicted as positive, then recall obtained is 100%, precision is 0%, F_1 is 0%, $\text{Acc}^M \approx 50\%$, and $\text{Acc}^\mu \approx 50\%$.

The evaluation processes were conducted with TF$\times$IDF and TF$\times$RF term weighting schemes. For each term weighting scheme, the evaluation processes were performed seven times with a different number of selected terms (10, 50,

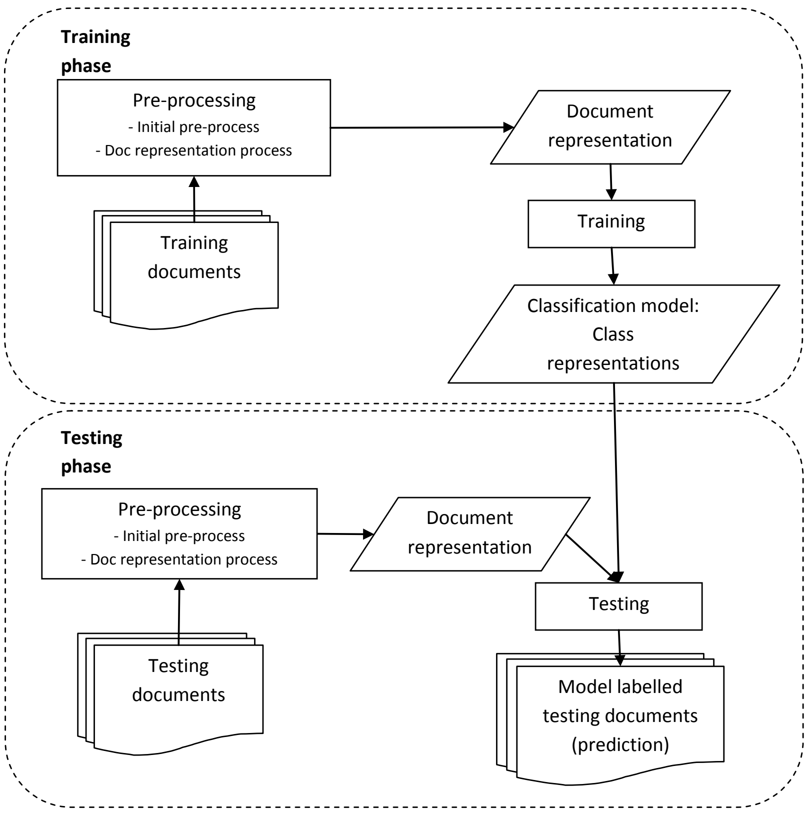


Figure 6.3: Text classification framework.

100, 150, 200, 250, and all terms). For TF×IDF term weighting the terms were selected based on DF; while for TF×RF term weighting the terms were selected based on TF×RF.

6.5.2 Results

This section presents the experiment results from comparing the proposed model RFD$_\mathcal{T}$ with the baseline models. Figures 6.4 to 6.11 [7] and Tables 6.10 and 6.11 present the results of the experiments using the TF×IDF and TF×RF term weighting. The performance of the RFD$\mathcal{T}$ model was based on an updated of decision

[7]Proposed model RFD$\mathcal{T}$ use 150 terms with TF×TDF term weighting scheme, however to make a clearer comparison with baseline models RFD$\mathcal{T}$ is also presented in these Figures.

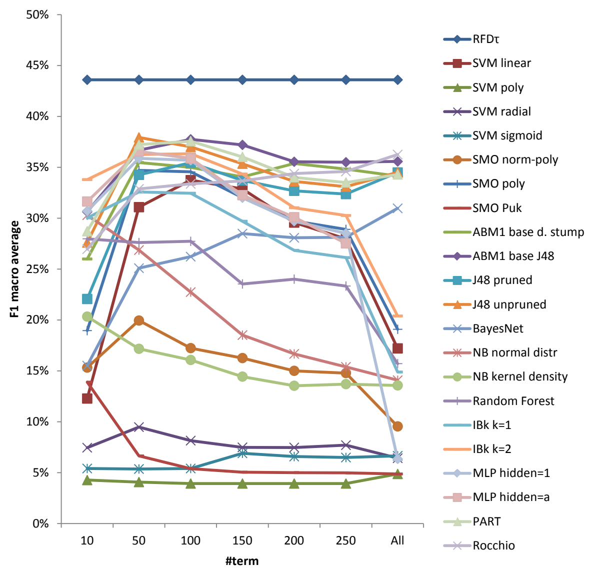


Figure 6.4: Experiment result with TF×IDF scheme for baselines: F1 macro average.

boundary (see Table 6.19). The best results for each baseline model are presented in Table 6.12 and Table 6.13. The values in bold represent the best results in a measurement, while the underlined values represent the results of the baseline models which were better than the proposed model. As shown in the results, the proposed model outperformed almost all the models in all measurements, except the micro-average F_1 for five models with TF×IDF and two models with TF×RF (indicated by underlining).

Table 6.14 shows that our proposed model outperforms Rough Set model in several threshold settings. Rough Set use 150 terms using TF×IDF term weighting scheme.

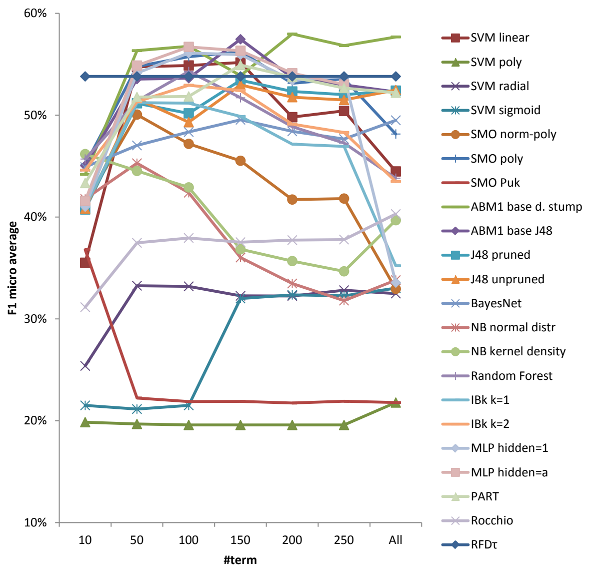


Figure 6.5: Experiment result with TF×IDF scheme for baselines: F1 micro average.

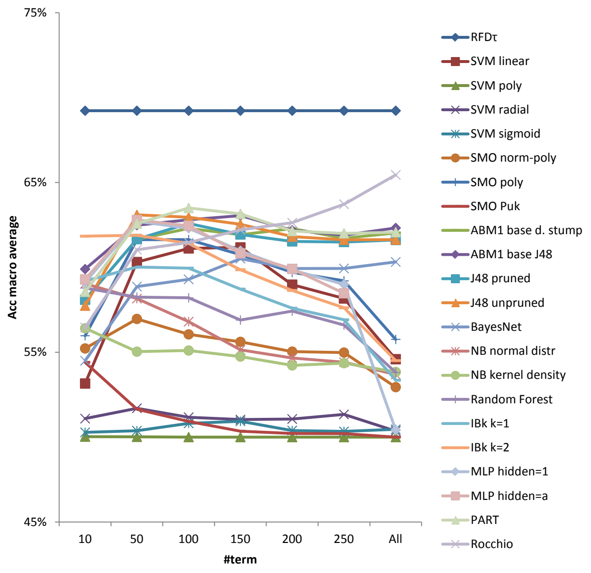


Figure 6.6: Experiment result with TF×IDF scheme for baselines: Accuracy macro average.

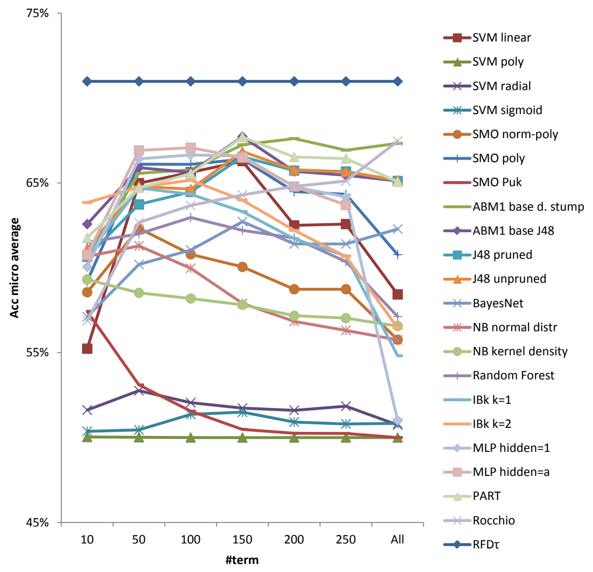


Figure 6.7: Experiment result with TF×IDF scheme for baselines: Accuracy micro average.

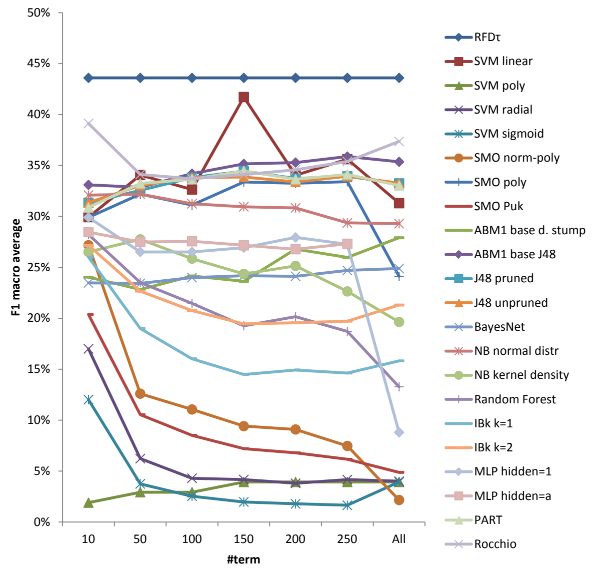


Figure 6.8: Experiment result with TF×RF scheme for baselines: F1 macro average.

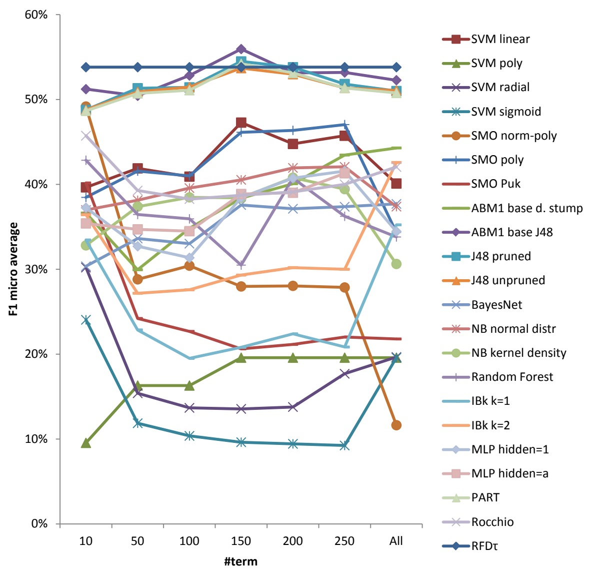


Figure 6.9: Experiment result with TF×RF scheme for baselines: F1 micro average.

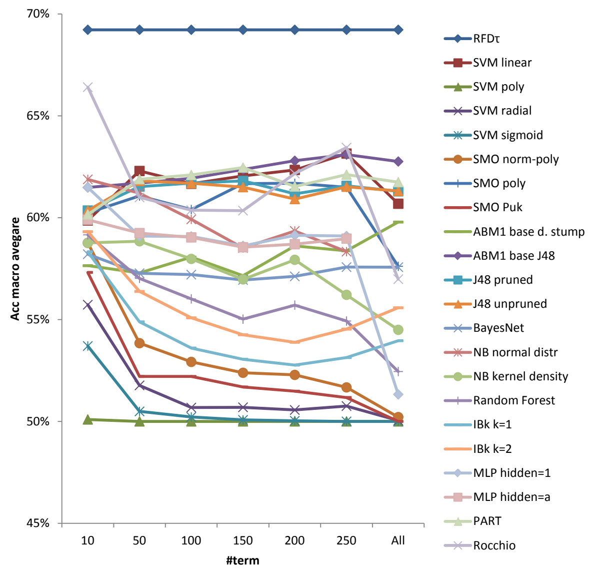


Figure 6.10: Experiment result with TF×RF scheme for baselines: Accuracy macro average.

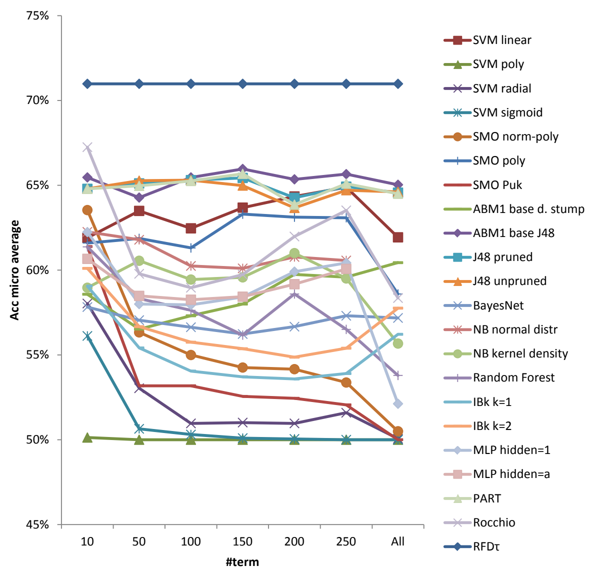


Figure 6.11: Experiment result with TF×RF scheme for baselines: Accuracy micro average.

Table 6.10: Experiment results with TF×IDF term weighting scheme for baseline models.

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$		0.428	0.688	0.537	0.711
SVM linear	10	0.123	0.532	0.355	0.552
	50	0.311	0.603	0.547	0.650
	100	0.337	0.611	0.549	0.656
	150	0.329	0.612	0.552	0.663
	200	0.296	0.590	0.498	0.625
	250	0.280	0.582	0.504	0.626
	All	0.172	0.546	0.445	0.584
SVM poly	10	0.043	0.500	0.199	0.500
	50	0.041	0.500	0.197	0.500
	100	0.039	0.500	0.196	0.500
	150	0.039	0.500	0.196	0.500
	200	0.039	0.500	0.196	0.500
	250	0.039	0.500	0.196	0.500
	All	0.049	0.500	0.218	0.500

Continued on next page

Table 6.10 – continued from previous page

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
SVM radial	10	0.075	0.511	0.254	0.516
	50	0.095	0.517	0.332	0.528
	100	0.081	0.512	0.332	0.521
	150	0.075	0.510	0.323	0.517
	200	0.075	0.511	0.322	0.516
	250	0.077	0.513	0.328	0.518
	All	0.064	0.504	0.325	0.507
SVM sigmoid	10	0.054	0.503	0.215	0.504
	50	0.054	0.504	0.211	0.504
	100	0.054	0.508	0.215	0.514
	150	0.069	0.509	0.320	0.515
	200	0.066	0.504	0.324	0.509
	250	0.065	0.503	0.323	0.508
	All	0.067	0.505	0.330	0.508
SMO norm-poly	10	0.153	0.552	0.413	0.586
	50	0.199	0.570	0.500	0.623
	100	0.172	0.560	0.472	0.608
	150	0.163	0.556	0.455	0.601
	200	0.150	0.550	0.417	0.587
	250	0.148	0.550	0.418	0.587
	All	0.095	0.529	0.329	0.558
SMO poly	10	0.190	0.560	0.451	0.592
	50	0.347	0.616	0.548	0.661
	100	0.345	0.616	0.557	0.661
	150	0.320	0.607	0.562	0.664
	200	0.297	0.597	0.531	0.645
	250	0.289	0.592	0.536	0.643
	All	0.191	0.558	0.481	0.608
SMO Puk	10	0.139	0.544	0.368	0.574
	50	0.066	0.516	0.222	0.531
	100	0.054	0.509	0.219	0.516
	150	0.050	0.503	0.219	0.505
	200	0.050	0.502	0.217	0.503
	250	0.050	0.502	0.219	0.502
	All	0.049	0.500	0.218	0.500
ABM1 base d. stump	10	0.260	0.581	0.442	0.606

Continued on next page

Table 6.10 – continued from previous page

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
	50	0.355	0.616	0.563	0.656
	100	0.350	0.623	0.568	0.658
	150	0.340	0.619	0.539	0.672
	200	0.354	0.623	0.580	0.676
	250	0.348	0.617	0.568	0.669
	All	0.341	0.620	0.577	0.673
ABM1 base J48	10	0.306	0.599	0.450	0.626
	50	0.367	0.625	0.535	0.659
	100	0.377	0.628	0.536	0.656
	150	0.372	0.630	0.574	0.677
	200	0.355	0.622	0.537	0.657
	250	0.355	0.619	0.530	0.654
	All	0.356	0.623	0.523	0.651
J48 pruned	10	0.221	0.581	0.407	0.606
	50	0.343	0.617	0.512	0.637
	100	0.354	0.626	0.502	0.645
	150	0.337	0.619	0.534	0.665
	200	0.327	0.615	0.523	0.657
	250	0.324	0.615	0.520	0.657
	All	0.345	0.616	0.524	0.651
J48 unpruned	10	0.276	0.577	0.408	0.611
	50	0.379	0.631	0.516	0.648
	100	0.370	0.630	0.493	0.646
	150	0.353	0.625	0.530	0.668
	200	0.336	0.618	0.518	0.658
	250	0.331	0.616	0.515	0.657
	All	0.345	0.616	0.525	0.652
BayesNet	10	0.155	0.545	0.449	0.571
	50	0.251	0.589	0.470	0.602
	100	0.262	0.593	0.483	0.610
	150	0.285	0.605	0.495	0.627
	200	0.281	0.599	0.484	0.614
	250	0.281	0.599	0.476	0.614
	All	0.310	0.603	0.495	0.623
NB normal distr	10	0.303	0.590	0.418	0.607
	50	0.269	0.582	0.453	0.613

Continued on next page

Table 6.10 – continued from previous page

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
	100	0.227	0.568	0.424	0.600
	150	0.185	0.551	0.360	0.579
	200	0.167	0.547	0.335	0.568
	250	0.154	0.544	0.318	0.563
	All	0.141	0.537	0.338	0.557
NB kernel density	10	0.203	0.564	0.462	0.593
	50	0.172	0.550	0.445	0.585
	100	0.161	0.551	0.429	0.582
	150	0.144	0.547	0.368	0.578
	200	0.135	0.542	0.357	0.572
	250	0.137	0.544	0.347	0.570
	All	0.136	0.538	0.397	0.566
Random Forest	10	0.280	0.588	0.457	0.615
	50	0.276	0.582	0.514	0.620
	100	0.277	0.582	0.543	0.630
	150	0.235	0.569	0.517	0.622
	200	0.240	0.574	0.489	0.618
	250	0.233	0.566	0.473	0.604
	All	0.157	0.538	0.438	0.571
IBk k=1	10	0.300	0.592	0.410	0.607
	50	0.326	0.600	0.512	0.647
	100	0.325	0.600	0.512	0.643
	150	0.297	0.587	0.499	0.633
	200	0.268	0.576	0.471	0.617
	250	0.261	0.569	0.469	0.607
	All	0.149	0.534	0.352	0.548
IBk k=2	10	0.338	0.618	0.446	0.638
	50	0.363	0.619	0.513	0.647
	100	0.363	0.614	0.529	0.652
	150	0.343	0.599	0.524	0.640
	200	0.310	0.586	0.491	0.622
	250	0.303	0.576	0.483	0.607
	All	0.204	0.545	0.435	0.565
MLP hidden=1	10	0.307	0.591	0.410	0.601
	50	0.359	0.627	0.541	0.664
	100	0.357	0.623	0.561	0.667

Continued on next page

Table 6.10 – continued from previous page

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
	150	0.320	0.610	0.559	0.666
	200	0.297	0.598	0.534	0.648
	250	0.285	0.590	0.538	0.642
	All	0.063	0.504	0.336	0.510
MLP hidden=a	10	0.316	0.593	0.416	0.608
	50	0.365	0.628	0.549	0.669
	100	0.359	0.624	0.567	0.671
	150	0.323	0.608	0.563	0.665
	200	0.301	0.599	0.541	0.648
	250	0.275	0.585	0.530	0.637
	All	NA	NA	NA	NA
PART	10	0.287	0.586	0.433	0.618
	50	0.372	0.626	0.518	0.647
	100	0.376	0.635	0.518	0.655
	150	0.360	0.631	0.549	0.677
	200	0.340	0.621	0.537	0.665
	250	0.335	0.620	0.527	0.664
	All	0.343	0.621	0.522	0.651
Rocchio	10	0.270	0.565	0.312	0.569
	50	0.329	0.610	0.375	0.627
	100	0.334	0.615	0.379	0.637
	150	0.337	0.622	0.375	0.643
	200	0.344	0.626	0.377	0.648
	250	0.346	0.637	0.378	0.651
	All	0.362	0.654	0.403	0.675

Table 6.11: Experiment results with TF×RF term weighting scheme for baseline models.

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$		0.428	0.688	0.537	0.711
SVM linear	10	0.299	0.599	0.396	0.619
	50	0.341	0.623	0.419	0.635
	100	0.326	0.617	0.409	0.625
	150	0.417	0.621	0.473	0.637
	200	0.340	0.623	0.448	0.643
	250	0.356	0.631	0.457	0.649
	All	0.313	0.607	0.401	0.619
SVM poly	10	0.019	0.501	0.095	0.501
	50	0.029	0.500	0.163	0.500
	100	0.029	0.500	0.163	0.500
	150	0.039	0.500	0.196	0.500
	200	0.039	0.500	0.196	0.500
	250	0.039	0.500	0.196	0.500
	All	0.039	0.500	0.196	0.500
SVM radial	10	0.170	0.557	0.302	0.580
	50	0.062	0.518	0.154	0.530
	100	0.043	0.507	0.137	0.510
	150	0.042	0.507	0.135	0.510
	200	0.038	0.506	0.138	0.510
	250	0.042	0.508	0.177	0.516
	All	0.040	0.500	0.197	0.501
SVM sigmoid	10	0.120	0.537	0.240	0.561
	50	0.037	0.505	0.119	0.506
	100	0.025	0.502	0.104	0.503
	150	0.020	0.501	0.096	0.501
	200	0.018	0.500	0.094	0.500
	250	0.016	0.500	0.092	0.500
	All	0.039	0.500	0.196	0.500
SMO norm-poly	10	0.272	0.588	0.492	0.635
	50	0.126	0.538	0.288	0.563
	100	0.110	0.529	0.304	0.550
	150	0.094	0.524	0.280	0.543
	200	0.091	0.523	0.280	0.542

Continued on next page

Table 6.11 – continued from previous page

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
	250	0.075	0.517	0.279	0.534
	All	0.022	0.502	0.116	0.505
SMO poly	10	0.299	0.602	0.385	0.616
	50	0.322	0.611	0.416	0.619
	100	0.311	0.604	0.410	0.613
	150	0.334	0.617	0.461	0.633
	200	0.333	0.617	0.464	0.631
	250	0.334	0.615	0.470	0.631
	All	0.241	0.576	0.343	0.586
SMO Puk	10	0.204	0.573	0.402	0.614
	50	0.105	0.522	0.242	0.532
	100	0.085	0.522	0.227	0.532
	150	0.072	0.517	0.206	0.526
	200	0.068	0.515	0.211	0.524
	250	0.062	0.512	0.220	0.521
	All	0.049	0.500	0.218	0.500
ABM1 base d. stump	10	0.240	0.576	0.366	0.586
	50	0.228	0.573	0.299	0.565
	100	0.241	0.581	0.348	0.573
	150	0.236	0.571	0.384	0.580
	200	0.268	0.586	0.401	0.597
	250	0.260	0.584	0.435	0.596
	All	0.279	0.598	0.443	0.604
ABM1 base J48	10	0.331	0.615	0.512	0.655
	50	0.329	0.617	0.504	0.643
	100	0.342	0.619	0.528	0.655
	150	0.351	0.624	0.559	0.660
	200	0.353	0.628	0.531	0.654
	250	0.359	0.631	0.532	0.657
	All	0.354	0.628	0.523	0.650
J48 pruned	10	0.313	0.603	0.488	0.648
	50	0.326	0.615	0.513	0.651
	100	0.338	0.617	0.514	0.653
	150	0.344	0.618	0.545	0.654
	200	0.337	0.612	0.538	0.643
	250	0.339	0.616	0.518	0.649

Continued on next page

Table 6.11 – continued from previous page

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
	All	0.332	0.613	0.510	0.646
J48 unpruned	10	0.313	0.603	0.488	0.648
	50	0.330	0.618	0.509	0.653
	100	0.338	0.617	0.515	0.653
	150	0.339	0.615	0.537	0.650
	200	0.334	0.609	0.530	0.637
	250	0.339	0.615	0.513	0.647
	All	0.333	0.613	0.511	0.646
BayesNet	10	0.235	0.582	0.303	0.578
	50	0.234	0.573	0.336	0.570
	100	0.240	0.572	0.330	0.566
	150	0.242	0.569	0.376	0.562
	200	0.241	0.571	0.371	0.567
	250	0.247	0.576	0.374	0.573
	All	0.249	0.576	0.377	0.572
NB normal distr	10	0.321	0.619	0.370	0.623
	50	0.322	0.612	0.381	0.618
	100	0.312	0.599	0.396	0.602
	150	0.309	0.585	0.405	0.601
	200	0.308	0.593	0.419	0.608
	250	0.294	0.583	0.421	0.606
	All	0.293	0.697	0.374	0.703
NB kernel density	10	0.265	0.588	0.328	0.590
	50	0.277	0.588	0.374	0.605
	100	0.258	0.580	0.385	0.594
	150	0.244	0.570	0.384	0.596
	200	0.251	0.579	0.408	0.610
	250	0.226	0.562	0.394	0.595
	All	0.196	0.545	0.306	0.557
Random Forest	10	0.282	0.592	0.428	0.614
	50	0.235	0.570	0.364	0.583
	100	0.215	0.560	0.359	0.576
	150	0.193	0.550	0.305	0.562
	200	0.202	0.557	0.407	0.586
	250	0.187	0.549	0.362	0.565
	All	0.133	0.524	0.338	0.538

Continued on next page

Table 6.11 – continued from previous page

Model	#Term	Macro-average		Micro-average	
		F_1^M	Acc^M	F_1^μ	Acc^μ
IBk k=1	10	0.260	0.583	0.334	0.591
	50	0.190	0.549	0.229	0.554
	100	0.160	0.536	0.195	0.540
	150	0.145	0.530	0.208	0.537
	200	0.149	0.528	0.224	0.536
	250	0.146	0.531	0.208	0.539
	All	0.158	0.540	0.352	0.562
IBk k=2	10	0.272	0.593	0.364	0.601
	50	0.227	0.564	0.272	0.567
	100	0.207	0.551	0.276	0.557
	150	0.194	0.543	0.293	0.554
	200	0.196	0.539	0.302	0.549
	250	0.197	0.545	0.300	0.554
	All	0.213	0.556	0.426	0.578
MLP hidden=1	10	0.299	0.615	0.372	0.622
	50	0.265	0.591	0.327	0.580
	100	0.265	0.591	0.313	0.579
	150	0.269	0.586	0.384	0.584
	200	0.279	0.591	0.407	0.599
	250	0.273	0.591	0.416	0.604
	All	0.088	0.513	0.344	0.521
MLP hidden=a	10	0.285	0.599	0.354	0.607
	50	0.275	0.592	0.347	0.585
	100	0.276	0.590	0.345	0.583
	150	0.272	0.585	0.389	0.584
	200	0.268	0.587	0.390	0.592
	250	0.273	0.590	0.413	0.601
	All	NA	NA	NA	NA
PART	10	0.309	0.602	0.487	0.648
	50	0.332	0.619	0.507	0.650
	100	0.336	0.621	0.511	0.653
	150	0.345	0.625	0.542	0.657
	200	0.336	0.615	0.531	0.639
	250	0.341	0.621	0.513	0.651
	All	0.330	0.617	0.508	0.645
Rocchio	10	0.391	0.664	0.457	0.672

Continued on next page

Table 6.11 – continued from previous page

Model	#Term	Macro-average F_1^M	Macro-average Acc^M	Micro-average F_1^μ	Micro-average Acc^μ
	50	0.341	0.610	0.393	0.598
	100	0.337	0.604	0.383	0.590
	150	0.341	0.603	0.387	0.597
	200	0.346	0.622	0.391	0.620
	250	0.354	0.634	0.400	0.635
	All	0.373	0.570	0.420	0.584

Test of Statistical Significance The *t-test p* values in Table 6.15, indicate that the results for the proposed RFD$_\tau$ model was statistically significant, except in six measurements for four models (indicated by underlining). It should be noted that although some models had measurements that outperformed the proposed model in the micro-average values (see underlined results in Table 6.12 and Table 6.13), the performances was still significantly lower than the proposed models. This is because the significance of the *t-test p* values measurements was affected more by the macro-average values.

Comparison with Proportional Decision Boundary Setting In the proportional decision boundary setting [175], it is assumed that $|D^+| : |D^-| \approx |U^+| : |U^-|$. Compared to proportional decision boundary setting, which is a popular decision boundary setting model, the proposed decision boundary setting model was equivalent (see Table 6.16). However in proportional decision boundary setting the number of testing dataset has to be known in advance, so it is not suitable for online testing.

Comparison with Tuned Decision Boundary Setting In tuned decision boundary setting [54], the decision boundary is set for each category in order to obtain

Table 6.12: Experiment results with TF×IDF term weight for baseline models (best performance).

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$	**0.428**	**0.688**	0.537	**0.711**
SVM linear	0.337	0.611	<u>0.549</u>	0.656
SVM poly	0.049	0.500	0.218	0.500
SVM radial	0.095	0.517	0.332	0.528
SVM sigmoid	0.069	0.509	0.320	0.515
SMO norm-poly	0.199	0.570	0.500	0.623
SMO poly	0.345	0.616	<u>0.557</u>	0.661
SMO Puk	0.139	0.544	0.368	0.574
ABM1 base d. stump	0.350	0.623	<u>**0.568**</u>	0.658
ABM1 base J48	0.377	0.628	0.536	0.656
J48 pruned	0.354	0.626	0.502	0.645
J48 unpruned	0.379	0.631	0.516	0.648
BayesNet	0.285	0.605	0.495	0.627
NB normal distr	0.303	0.590	0.418	0.607
NB kernel density	0.203	0.564	0.462	0.593
Random Forest	0.280	0.588	0.457	0.615
IBk k=1	0.326	0.600	0.512	0.647
IBk k=2	0.363	0.619	0.513	0.647
MLP hidden=1	0.359	0.627	<u>0.541</u>	0.664
MLP hidden=a	0.365	0.628	<u>0.549</u>	0.669
PART	0.376	0.635	0.518	0.655
Rocchio	0.362	0.654	0.403	0.675

Table 6.13: Experiment results with TF×RF term weight for baseline models (best performance).

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$	**0.428**	**0.688**	0.537	**0.711**
SVM linear	0.417	0.621	0.473	0.637
SVM poly	0.039	0.500	0.196	0.500
SVM radial	0.170	0.557	0.302	0.580
SVM sigmoid	0.120	0.537	0.240	0.561
SMO norm-poly	0.272	0.588	0.492	0.635
SMO poly	0.334	0.617	0.461	0.633
SMO Puk	0.204	0.573	0.402	0.614
ABM1 base d. stump	0.279	0.598	0.443	0.604
ABM1 base J48	0.359	0.631	0.532	0.657
J48 pruned	0.344	0.618	**<u>0.545</u>**	0.654
J48 unpruned	0.338	0.617	0.515	0.653
BayesNet	0.242	0.569	0.376	0.562
NB normal distr	0.322	0.612	0.381	0.618
NB kernel density	0.277	0.588	0.374	0.605
Random Forest	0.282	0.592	0.428	0.614
IBk k=1	0.260	0.583	0.334	0.591
IBk k=2	0.272	0.593	0.364	0.601
MLP hidden=1	0.299	0.615	0.372	0.622
MLP hidden=a	0.285	0.599	0.354	0.607
PART	0.345	0.625	<u>0.542</u>	0.657
Rocchio	0.391	0.664	0.457	0.672

Table 6.14: Experiment results for Rough Set.

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$	**0.428**	**0.688**	**0.537**	**0.711**
RS min(min$_P$, max$_N$)	0.335	0.608	0.397	0.643
RS ave	0.176	0.546	0.305	0.576
RS min$_P$	0.335	0.608	0.397	0.643
RS max$_N$	0.083	0.527	0.203	0.548
RS proportional	0.291	0.589	0.419	0.616

Table 6.15: p-values for all models with TF×IDF and TF×RF term weighting (best performance)comparing with RFD$_\tau$ model in all accessing topics.

Model	TF×IDF		TF×RF	
	F_1	Acc	F_1	Acc
SVM linear	6.7E-05	9.5E-07	<u>7.7E-01</u>	1.5E-04
SVM poly	4.7E-16	1.5E-15	4.7E-16	1.5E-15
SVM radial	2.5E-14	2.1E-13	9.8E-11	5.6E-10
SVM sigmoid	1.7E-15	6.4E-15	2.2E-15	5.7E-13
SMO norm-poly	2.4E-11	8.7E-09	3.6E-06	3.3E-06
SMO poly	2.7E-04	1.9E-05	2.7E-03	6.2E-05
SMO Puk	1.2E-11	1.9E-09	2.1E-09	2.8E-07
ABM1 base d. stump	6.1E-03	9.5E-04	4.4E-04	1.2E-04
ABM1 base J48	<u>8.2E-02</u>	2.6E-03	4.0E-02	1.8E-02
J48 pruned	9.0E-03	2.7E-03	9.6E-03	1.5E-03
J48 unpruned	<u>7.4E-02</u>	6.9E-03	5.7E-03	1.1E-03
BayesNet	3.3E-05	8.0E-06	1.7E-05	3.7E-07
NB normal distr	2.3E-05	3.9E-06	5.3E-05	1.7E-05
NB kernel density	1.1E-08	8.5E-08	1.4E-06	2.7E-07
Random Forest	4.3E-06	1.8E-06	3.7E-05	1.6E-06
IBk k=1	2.4E-05	4.8E-07	2.0E-05	6.4E-07
IBk k=2	3.8E-03	3.7E-05	1.9E-04	1.3E-05
MLP hidden=1	4.6E-03	3.6E-04	6.0E-04	5.7E-04
MLP hidden=a	9.9E-03	4.9E-04	2.1E-04	6.1E-05
PART	4.8E-02	9.4E-03	1.3E-02	4.5E-03
Rocchio	9.4E-03	<u>5.8E-02</u>	<u>1.8E-01</u>	<u>2.4E-01</u>

Table 6.16: Experiment result for Propotional Decision Boundary Setting.

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$	**0.428**	**0.688**	**0.537**	**0.711**
RFD$_{prop}$	0.427	0.676	0.527	0.688

Table 6.17: Experiment result for Tuned Decision Boundary Setting.

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$	**0.428**	**0.688**	**0.537**	**0.711**
RFD$_{tuned}$	0.339	0.630	0.501	0.650

Table 6.18: Improving performance using positive, negative, and general vectors in uncertain boundary.

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_\tau$ (using 3 vectors)	**0.436**	**0.692**	**0.538**	**0.790**
RFD$_{\tau'}$	0.427	0.676	0.527	0.688

the highest training performance. In this decision boundary setting, the highest performance for training is look for exhaustively from the lowest to the highest training document's weight. The results presented in Table 6.17 indicate that the proposed decision boundary setting was more effective. Furthermore, the proposed decision boundary setting was more efficient compared to the tuned decision boundary setting that had to be set for every category.

Improving Performance Using Positive, Negative, and General Vectors in Uncertain Boundary As shown in Table 6.18, the use of specific and generic vectors improved the initial decision boundary RFD$_{\tau'}$ performance. The explanation of RFD$_{\tau'}$ in Section 4.3.1.

6.5.3 Discussion

Updating Initial Decision Boundary After the boundary region, and the initial decision boundary were set, the decision boundary was adjusted. Table 6.19 shows the alternative versions of decision boundary update, where τ' is the ini-

Table 6.19: Update initial decision boundary.

Variant	Description
RFD$_{\tau'}$	$\tau' = min(\tau_N, \tau_P)$
RFD$_\tau v1$	if $\tau_N < \tau_P$ then $\tau = \tau' - 0.1(\|\tau_P - \tau_N\|)$
	if $\tau_N > \tau_P$ then $\tau = \tau' + 0.1(\|\tau_P - \tau_N\|)$
RFD$_\tau v2$	$\tau = \tau' + 0.1(\|\tau_P - \tau_N\|)$
RFD$_\tau v3$	$\tau = \tau' - 0.1(\|\tau_P - \tau_N\|)$
RFD$_\tau v4$	if $\tau_N < \tau_P$ then $\tau = \tau' - 0.1(\|\tau_P - \tau_N\|)$
	if $\tau_N > minP$ then $\tau = \tau' + 0.2(\|\tau_P - \tau_N\|)$
RFD$_\tau v5$	if $\tau_N < \tau_P$ then $\tau = \tau'$
	if $\tau_N > \tau_P$ then $\tau = \tau' + 0.1(\|\tau_P - \tau_N\|)$

Table 6.20: RFD$_\tau$ update initial decision boundary.

Variant	Macroaverage		Microaverage	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_{\tau'}$	0.426	0.682	0.527	0.701
RFD$_\tau v1$	**0.428**	**0.688**	**0.537**	**0.711**
RFD$_\tau v2$	0.427	0.680	0.532	0.702
RFD$_\tau v3$	0.429	0.686	0.521	0.701
RFD$_\tau v4$	0.425	0.683	0.538	0.707
RFD$_\tau v5$	0.428	0. 683	0.535	0.707

tial decision boundary. The best performance was reached by RFD$_\tau v1$ (see Table 6.20).

Decision Boundary for Optimal Classification Performance In section 4.2, it is stated that classification effectiveness will be better around the middle of the document weight distribution where the boundary region most probably located. This paragraph shows an analysis of that statement.

Let $\tau_{min} < \tau_{low'} < \tau_{high'} < \tau_{max}$, where $\tau_{min} = \min_{d\in U}\{score(d)\}$ $\tau_{max} = \max_{d\in U}\{score(d)\}$. Then high performance (in term of F$_1$ and Accuracy) of classification models have maximum performance with decision boundary $\tau_{low'} < \tau < \tau_{high'}$. Where $\tau_{low'} > \min_{d\in D}\{score(d)\}$, and $\tau_{high'} < \max_{d\in D}\{score(d)\}$

Assume

Let $d_i \in \{d|U^+\}$, and $d_j \in \{d|U^-\}$, then:

- $model_p$ is a perfect scoring model $\iff prob(score(d_i) \geq score(d_j)) = 1$. This model represents ideal performance (in term of F_1 and Accuracy) of classification model.
- $model_r$ is a random scoring model $\iff prob(score(d_i) \geq score(d_j)) \approx 0.5$.
- $model_g$ is a good scoring model $\iff prob(score(d_i) \geq score(d_j)) \gg 0.5$. This model represents high performance (in term of F_1 and Accuracy) of classification model.

Let τ is decision boundary for classification. Let τ_{ideal} is a decision boundary for a perfect model where all document are correctly predicted; so performance will maximum at τ_{ideal}.

Let TP_τ, FP_τ, TN_τ, FN_τ, $R_\tau = TP/|U^+|$, $P_\tau = TP/(TP+FP)$, $F_{1_\tau} = \frac{2\times(R\times P)}{R+P}$, $Acc_\tau = \frac{TP+TN}{U}$ be true positive, false positive, true negative, false negative, recall, recall, precision, F_1, and accuracy with decision boundary τ[8].

Let $|U^+| \approx |U^-|$[9]. With τ = τ_{min} we found $TP_{\tau_{min}} = |U^+|$, $FP_{\tau_{min}} = |U^-|$, $TN_{\tau_{min}} = 0$, $FN_{\tau_{min}} = 0$. $R_{\tau_{min}} = 1$, $P_{\tau_{min}} = U^+/(U^+ + U^-) \approx 0.5$, $F_{1_{\tau_{min}}} \approx 0.67$, $Acc_{\tau_{min}} \approx 0.5$.

With τ = τ_{max} we found $TP_{\tau_{max}} = 0, FP_{\tau_{max}} = 0, TN_{\tau_{max}} = |U^-|, FN_{\tau_{max}} = |U^+|$. $R_{\tau_{max}} = 0$, $P_{\tau_{max}} = 0) \approx 0.5$, $F_{1_{\tau_{min}}} = 0$, $Acc_{\tau_{max}} \approx 0.5$.

If we move τ from τ_{min} to τ_{max}, we can find that,

- For $model_p$:

[8]These values are described in more detail in Chapter 6

[9]To ensure the accuracy measurement does not mislead, a more detailed explanation is provided in Chapter 6

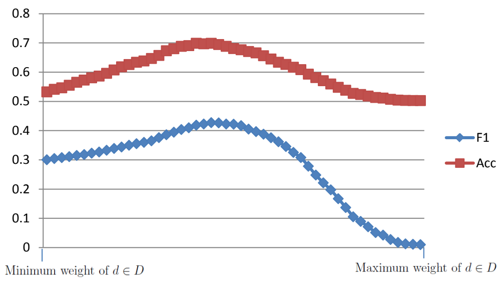
- TP steady at $|U^+|$ from τ_{min} until at around τ_{ideal} and then decrease sharply to 0 at τ_{max}.
- FP decrease sharply from $|U^-|$ at τ_{min} to 0 at around τ_{ideal} and then steady until τ_{max}.
- TN increase sharply from 0 at τ_{min} to $|U^-|$ at around τ_{ideal} and then steady until at τ_{max}.
- FN steady 0 from τ_{min} until at around τ_{ideal} and then increase sharply to U^+ at τ_{max}.
- R steady 1 from τ_{min} until at around τ_{ideal} and then decrease sharply to 0 at τ_{max}.
- P increase from ~0.5 at τ_{min} to 1 at around τ_{ideal} and then decrease smoothly except near the end to 0 at τ_{max}.
- F_1 increase from 0.67 at τ_{min} to 1 at around τ_{ideal} and decrease to 0 at τ_{max}.
- Acc increase from ~0.5 at τ_{min} to 1 at around τ_{ideal} and then decrease to 0.5 at τ_{max}.

* For $model_r$:
  - TP decrease from $|U^+|$ at τ_{min} to ~$\frac{|U^+|}{2}$ at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
  - FP decrease from $|U^-|$ at τ_{min} to ~$\frac{|U^-|}{2}$ at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
  - TN increase from 0 at τ_{min} to ~$\frac{|U^-|}{2}$ at around τ_{ideal} and then continue increase to U^- at τ_{max}.
  - FN increase from 0 at τ_{min} to ~$\frac{|U^+|}{2}$ at around τ_{ideal} and then continue increase to U^+ at τ_{max}.

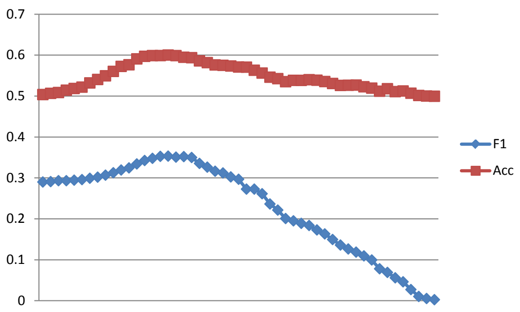
- R decrease from 1 at τ_{min} to ~0.5 at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
- P steady ~0.5 from τ_{min} until at around τ_{ideal} and then continue steady at ~0.5 until τ_{max}.
- F_1 decrease from 0.67 at τ_{min} to ~0.5 at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
- Acc steady ~0.5 from τ_{min} until at around τ_{ideal} and then continue steady at ~0.5 until τ_{max}.

• For $model_g$:

- TP decrease from $|U^+|$ at τ_{min} to $|U^+| \geq TP_{\tau_{ideal}} \geq$ ~$\frac{|U^+|}{2}$ at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
- FP decrease from $|U^-|$ at τ_{min} to $0 \leq FP_{\tau_{ideal}} \leq$ ~$\frac{|U^-|}{2}$ at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
- TN increase from 0 at τ_{min} to ~$\frac{|U^-|}{2} \leq TN_{\tau_{ideal}} \leq |U^-|$ at around τ_{ideal} and then continue increase to U^- at τ_{max}.
- FN increase from 0 at τ_{min} to $0 \leq FN_{\tau_{ideal}} \leq$ ~$\frac{|U^+|}{2}$ at around τ_{ideal} and then continue increase to U^+ at τ_{max}.
- R decrease from 1 at τ_{min} to ~$0.5 \leq R_{\tau_{ideal}} \leq 1$ at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
- P increase from ~0.5 at τ_{min} to ~$0.5 \leq P_{\tau_{ideal}} \leq 1$ at around τ_{ideal} and then decrease to 0 at τ_{max}.
- F_1 increase from 0.67 at τ_{min} to ~$0.5 \leq F_{1_{\tau_{ideal}}} \leq 1$ at around τ_{ideal} and then continue decrease to 0 at τ_{max}.
- Acc increase from ~0.5 at τ_{min} to ~$0.5 \leq Acc_{\tau_{ideal}} \leq 1$ at around τ_{ideal} and then decrease to ~0.5 at τ_{max}.



(a) RFD



(b) Rocchio

Figure 6.12: Macro average of RFD$_\tau$ and Rocchio performance at different decision boundaries.

F_1 and Acc trends for $model_g$shows that high performance (in term of F$_1$ and Accuracy) of classification models have maximum performance with decision boundary $\tau_{low'} < \tau < \tau_{high'}$.

Figure 6.12 shows the experimentally global optimum for macro-average RFD and Rocchio model.

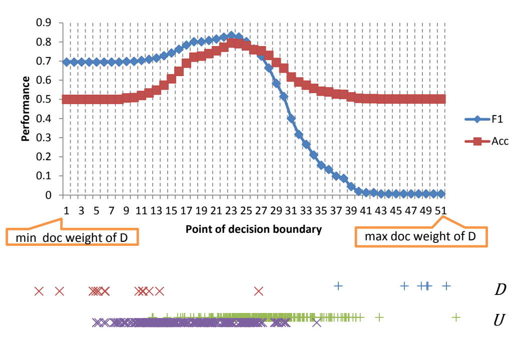
Table 6.21 shows the location of the maximum F$_1$ with TF$\times$IDF term weighting for all topics. It can be seen that the maximum F$_1$ was inside the decision boundary region in 25 topics, while the maximum F$_1$ was near the decision boundary region in 9 topics.

Figure 6.13 shows the results topic 1 and topic 15 using the RFD$_\tau$. In topic 1, the maximum performance was outside the decision boundary region; while in topic 15 the maximum performance was inside the decision boundary region. The figures for all the topics using the RFD model are presented in Appendix C.

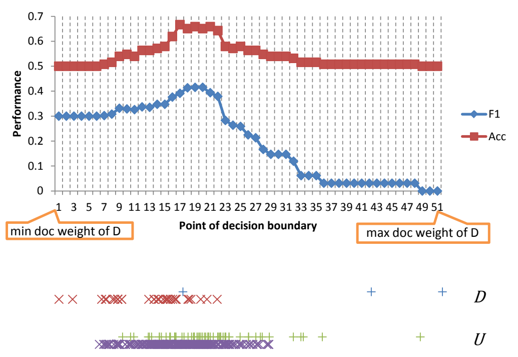
In the Rocchio model, the decision boundary is zero value, which is represented with the diamond symbol; for example, see Figure 6.14. In this figure, it can seen that F$_1$ in topic 1 did not reach maximum, while in topic 40 F$_1$ was the maximum for that topic. The figures for all the topics using the Rocchio model are presented in Appendix D.

Table 6.21: Maximum of F_1 RFD_τ and Region Boundary Region.

Topic	min	max	max F1	point of min	point of max	point of max F1	maximum F_1
1	1.044	1.741	0.834	27	37	22	
2	0.859	5.578	0.820	5	44	11	Inside
3	1.080	1.140	0.737	22	23	21	Near
4	2.510	5.568	0.697	16	42	33	Inside
5	1.702	4.013	0.656	21	36	22	Inside
6	3.600	5.912	0.252	21	36	17	
7	1.214	2.368	0.400	28	34	33	Inside
8	4.806	7.600	0.533	28	44	25	Near
9	1.017	5.972	0.526	4	44	7	Inside
10	2.601	2.859	0.508	25	27	29	Near
11	3.011	11.414	0.357	20	50	18	Near
12	2.926	3.461	0.513	30	35	32	Inside
13	3.008	3.510	0.444	23	28	29	Near
14	1.053	1.942	0.437	16	30	31	Near
15	1.084	1.535	0.416	16	21	19	Inside
16	1.283	1.730	0.740	28	35	21	
17	2.472	3.669	0.667	14	28	18	Inside
18	2.611	4.087	0.208	30	47	20	
19	6.227	6.734	0.557	44	50	36	
20	1.233	1.541	0.701	31	40	32	Inside
21	3.884	4.305	0.698	24	28	26	Inside
22	1.928	2.052	0.780	22	23	25	Near
23	4.196	4.286	0.480	38	38	39	
24	2.466	2.918	0.259	30	36	22	
25	3.256	5.320	0.540	23	41	22	Near
26	3.081	3.823	0.912	29	41	22	
27	1.670	2.385	0.481	31	39	31	Inside
28	2.473	3.442	0.404	21	29	27	Inside
29	3.500	5.862	0.448	25	42	28	Inside
30	4.249	8.563	0.429	18	43	24	Inside
31	1.955	4.912	0.727	13	35	20	Inside
32	1.881	3.577	0.171	21	43	34	Inside
33	2.505	3.348	0.538	29	40	30	Inside
34	0.418	0.998	0.321	21	31	6	
35	3.075	3.765	0.890	16	28	17	Inside
36	3.646	4.248	0.367	32	38	26	
37	2.142	4.308	0.560	16	32	28	Inside
38	4.161	4.534	0.364	27	32	21	
39	4.599	7.369	0.647	29	47	37	Inside
40	0.331	0.652	0.487	33	39	34	Inside
41	2.020	3.742	0.598	21	39	22	Inside
42	0.651	1.459	0.368	12	33	19	Inside
43	5.628	5.633	0.184	33	33	21	
44	2.384	2.477	0.512	30	31	18	
45	1.168	2.525	0.157	19	39	28	Inside
46	2.558	8.320	0.633	14	47	3	
47	1.503	2.004	0.460	30	37	34	Inside
48	1.348	1.620	0.896	26	29	23	Near
49	0.884	1.079	0.247	15	17	9	
50	1.372	1.752	0.377	41	46	30	

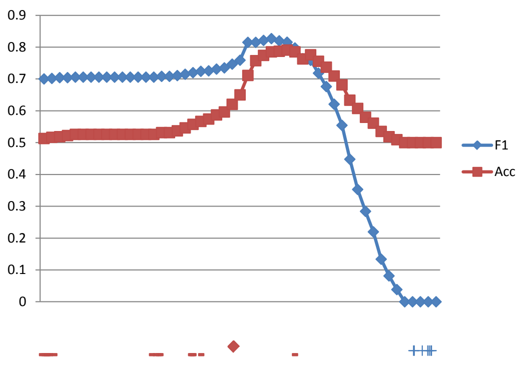


(a) Topic 1

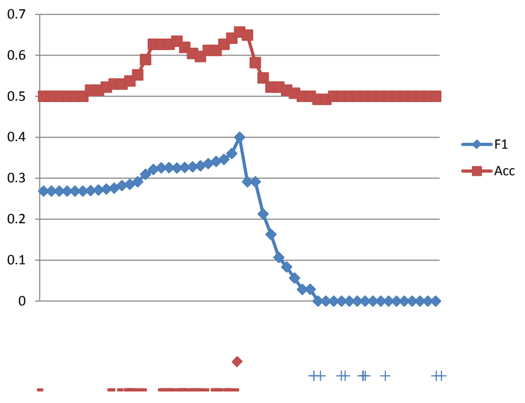


(b) Topic 15

Figure 6.13: RFD$_{\tau}$ Performance at different Decision Boundaries.



(a) Topic 1



(b) Topic 40

Figure 6.14: Rocchio Performance at Different Decision Boundaries.

Table 6.22: Maximal performance of RFD$_\tau$ and Rocchio models.

Model	F_1^M	Acc^M
RFD$_\tau$ (max performance)	0.519	0.758
RFD$_\tau$	0.428	0.688
Rocchio (max performance)	0.463	0.722
Rocchio	0.362	0.654

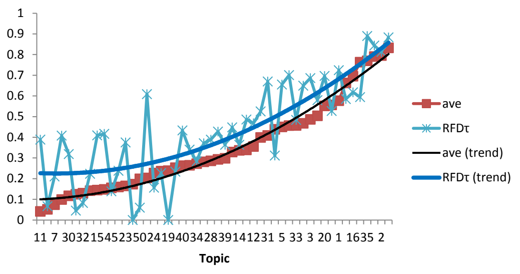
Table 6.23: The best baseline models.

No	Model	No of term
1	SVM linear	100 term
2	SMO poly	100 terms
3	ABM1 base J48	100 terms
4	J48 unprunned	50 terms
5	NB normal distr	10 terms
6	Random Forest	10 terms
7	IBk k=2	50 terms
8	MLP hidden = a	50 terms
9	PART	100 terms
10	Rocchio	all terms

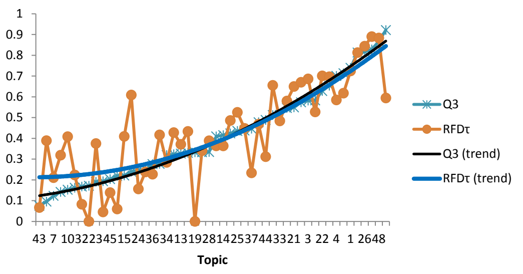
The results in Table 6.22 shows that the maximum performance for the RFD$_\tau$ and Rocchio models was much higher than the current setting of the RFD$_\tau$ and Rocchio. The maximum performance was macro-average of the best performances in all topics. The results in Table 6.22 indicates that the performance can still be increased.

In summary, it shows the potential of the proposed decision boundary setting model in choosing the optimal performance.

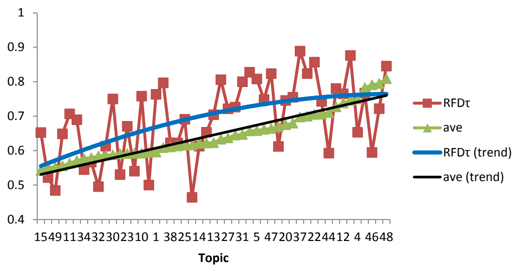
RFD$_\tau$ Performance in Classification Difficulty The classification difficulty of a topic can be estimated from the performance of comprehensive types of classifiers. From the baseline models (see Table 6.4) models with the best macro-average performance for each type were selected (see Table 6.23).



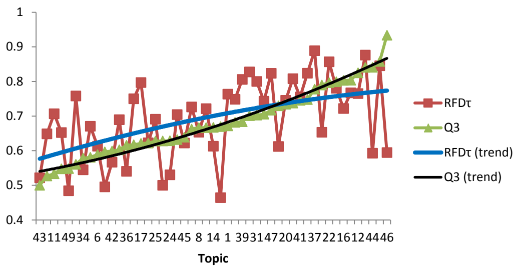
(a) F_1 sorted by ave



(b) F_1 sorted by Q3



(c) Acc sorted by ave



(d) Acc sorted by Q3

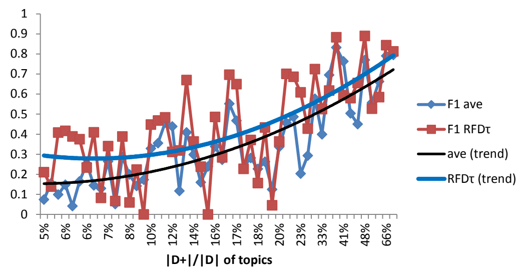
Figure 6.15: RFD_τ performance over topic difficulty.

The average and quartile three (Q3) performance for the models in Table 6.23 was selected to indicate relative difficulty of the topics. The TF×IDF term weighting scheme was used. The result, as illustrated in Figure 6.15 shows that the RFD_τ model had better performance than the baseline models especially in relation to the classification of difficult topics. The trendlines in this figure and the following figures were generated by the polynomial order two of the trend/regression type. Figure 6.16 shows the results in more detail for F_1. In 28 topics, RFD_τ had better F_1 performance than the Q3 of baseline models.

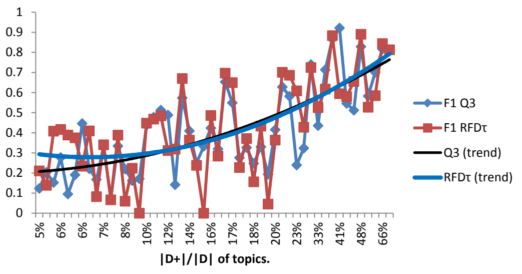
RFD_τ vs. Baseline Models Performance in Training Set Imbalance Rate
Figure 6.17 presents the results of the comparison of the performance of the RFD_τ and baseline models in training dataset imbalance rate. Figure 6.18 shows more detail for Q3. As the figures show, the RFD_τ outperforms baseline models especially on a highly imbalanced of training dataset.

Topic	RFDτ	SVM	SMO	ABM1	J48	NB	RForest	IBk	MLP	PART	Rocchio	Q3
43	0.056	0.000	0.000	0.064	0.070	🟩 0.151	0.071	0.000	0.000	🟩 0.083	🟩 0.077	0.075
11	🟩 0.357	0.000	0.000	🟩 0.095	🟩 0.095	0.000	0.000	0.000	0.000	🟩 0.095	🟩 0.130	0.095
7	🟩 0.188	🟩 0.133	0.093	0.000	0.000	0.043	0.000	🟩 0.212	0.077	0.045	🟩 0.137	0.123
30	🟩 0.333	0.100	🟩 0.261	0.080	0.044	🟩 0.143	0.104	0.093	🟩 0.162	0.044	0.139	0.142
10	🟩 0.339	0.000	0.000	🟩 0.205	0.115	0.054	0.000	🟩 0.165	0.042	0.043	🟩 0.366	0.153
6	🟩 0.214	0.108	0.056	0.053	🟩 0.162	🟩 0.243	0.150	0.161	0.133	0.093	🟩 0.260	0.162
32	0.119	0.067	0.118	0.000	0.138	🟩 0.168	0.074	🟩 0.190	0.113	🟩 0.262	0.167	0.168
18	0.000	0.000	0.000	🟩 0.727	🟩 0.476	0.000	0.063	0.062	🟩 0.190	0.100	0.115	0.172
23	🟩 0.529	0.167	🟩 0.286	🟩 0.205	0.100	0.111	0.095	0.178	0.160	🟩 0.195	0.153	0.191
49	0.065	0.000	0.028	0.133	0.070	🟩 0.198	0.176	0.180	0.048	🟩 0.203	🟩 0.210	0.193
45	0.135	0.000	0.061	0.140	🟩 0.316	0.050	0.108	0.168	🟩 0.215	0.150	🟩 0.348	0.204
50	0.063	0.174	0.194	0.212	0.086	🟩 0.222	0.116	🟩 0.333	0.135	0.145	🟩 0.375	0.220
15	🟩 0.368	0.030	0.067	0.000	🟩 0.225	0.210	0.031	0.202	0.135	🟩 0.225	🟩 0.324	0.221
17	🟩 0.609	0.211	🟩 0.244	0.042	0.226	🟩 0.301	0.098	🟩 0.294	0.197	0.226	0.195	0.240
24	0.164	0.197	0.222	0.222	🟩 0.253	0.214	🟩 0.289	0.231	0.192	🟩 0.308	0.142	0.248
42	🟩 0.253	0.000	0.077	0.000	🟩 0.514	0.067	0.167	🟩 0.290	🟩 0.258	0.000	0.226	0.250
36	0.268	0.141	0.101	0.184	🟩 0.397	🟩 0.325	0.269	0.183	0.212	0.274	🟩 0.276	0.276
8	🟩 0.571	0.000	0.000	🟩 0.303	🟩 0.303	0.200	0.000	0.186	0.069	🟩 0.303	0.116	0.277
34	0.300	0.311	0.193	🟩 0.322	0.224	0.303	0.159	0.223	🟩 0.370	0.256	🟩 0.364	0.319
29	🟩 0.426	0.304	🟩 0.385	0.288	0.224	🟩 0.330	0.214	🟩 0.328	0.310	0.258	0.290	0.324
13	🟩 0.373	0.325	0.286	0.252	🟩 0.341	0.226	0.135	0.309	0.256	🟩 0.327	🟩 0.357	0.326
40	🟩 0.497	0.262	0.315	0.136	🟩 0.344	0.216	0.105	0.294	🟩 0.335	0.219	🟩 0.402	0.330
19	0.000	0.113	0.300	0.169	0.098	0.214	0.316	🟩 0.352	🟩 0.338	0.098	🟩 0.386	0.333
38	0.290	0.319	🟩 0.369	0.250	0.214	0.062	0.077	0.324	🟩 0.368	0.329	🟩 0.335	0.333
28	🟩 0.397	0.204	0.204	🟩 0.341	0.308	0.313	0.172	🟩 0.364	0.289	🟩 0.405	0.262	0.334
39	0.350	🟩 0.435	0.286	0.296	0.176	🟩 0.519	🟩 0.452	0.333	0.105	0.253	0.129	0.409
14	0.364	0.380	🟩 0.427	0.258	0.194	0.376	0.237	🟩 0.444	🟩 0.485	0.208	0.359	0.415
27	🟩 0.495	0.354	0.384	🟩 0.465	0.281	0.286	0.148	0.229	0.341	🟩 0.465	🟩 0.437	0.424
25	🟩 0.527	0.416	0.354	0.347	🟩 0.470	0.422	🟩 0.440	0.375	0.296	0.419	🟩 0.454	0.436
47	🟩 0.452	🟩 0.526	🟩 0.515	0.393	0.250	0.154	0.042	0.418	0.257	🟩 0.444	0.287	0.438
37	0.247	0.333	0.000	🟩 0.462	🟩 0.462	0.000	0.125	0.182	0.400	🟩 0.462	0.105	0.446
12	0.379	0.333	0.462	🟩 0.606	🟩 0.483	0.179	0.189	0.242	0.364	🟩 0.596	0.100	0.477
44	0.395	0.359	0.385	🟩 0.500	🟩 0.694	0.212	0.239	0.447	0.425	🟩 0.671	0.454	0.488
5	🟩 0.661	0.416	0.352	0.479	0.500	🟩 0.578	🟩 0.516	0.257	0.336	🟩 0.600	0.469	0.512
33	0.500	🟩 0.558	0.488	🟩 0.513	🟩 0.513	0.383	0.408	0.358	🟩 0.583	🟩 0.513	0.256	0.513
41	🟩 0.595	0.545	🟩 0.638	0.457	0.436	0.456	0.512	🟩 0.546	🟩 0.624	0.384	0.441	0.546
21	🟩 0.685	0.535	0.497	🟩 0.738	0.546	0.262	0.183	0.446	🟩 0.585	🟩 0.550	0.333	0.549
31	🟩 0.674	0.550	0.528	0.500	🟩 0.581	0.123	0.051	0.464	🟩 0.618	0.071	🟩 0.602	0.574
3	🟩 0.600	0.333	0.346	0.577	🟩 0.707	0.250	🟩 0.583	0.519	0.521	🟩 0.719	0.312	0.582
9	0.523	0.576	0.545	🟩 0.583	0.416	🟩 0.639	🟩 0.696	0.494	0.581	0.508	0.530	0.583
22	🟩 0.774	🟩 0.635	0.608	🟩 0.693	0.306	0.168	0.318	0.418	0.441	🟩 0.694	0.286	0.628
20	🟩 0.685	🟩 0.683	🟩 0.659	0.519	0.635	0.412	0.302	0.542	0.478	0.505	🟩 0.785	0.653
4	0.587	0.667	0.691	🟩 0.728	🟩 0.704	0.627	0.598	0.604	0.685	🟩 0.751	0.573	0.701
16	0.628	0.615	0.642	🟩 0.715	0.709	🟩 0.760	0.682	🟩 0.745	0.702	0.709	0.675	0.713
1	0.683	🟩 0.740	0.686	0.436	0.426	0.354	0.482	🟩 0.742	0.738	0.411	🟩 0.754	0.739
2	🟩 0.814	🟩 0.814	0.799	🟩 0.821	0.804	0.730	0.788	0.812	0.787	🟩 0.844	0.753	0.813
26	🟩 0.885	0.799	0.805	0.713	0.745	0.799	🟩 0.866	🟩 0.821	0.799	0.730	🟩 0.828	0.817
35	🟩 0.893	🟩 0.832	0.817	0.790	0.787	0.563	0.573	🟩 0.864	0.819	0.790	🟩 0.864	0.828
48	🟩 0.892	0.615	0.821	🟩 0.892	0.830	🟩 0.867	0.826	0.859	🟩 0.918	0.830	0.865	0.867
46	0.594	0.651	0.682	🟩 0.967	🟩 0.967	0.664	0.742	0.646	0.783	🟩 0.967	0.563	0.921
No of topic with F1 >= Q3	28	9	9	19	19	14	7	15	14	22	22	

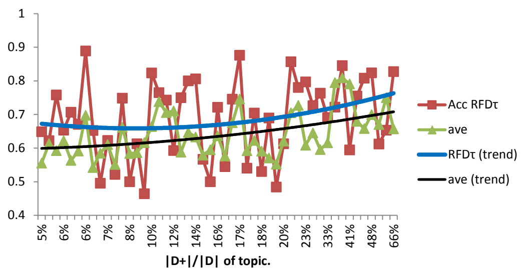
Figure 6.16: Visualisation of comparison of F_1 RFD_τ vs. baseline models sorted by Q3. Shaded numbers means higher than Q3.



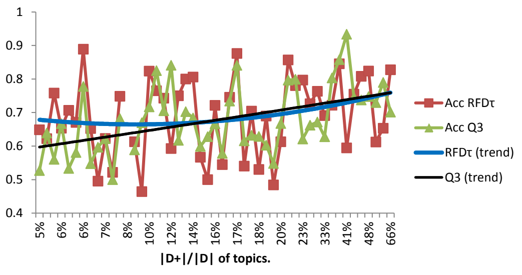
(a) Compare to average F_1.



(b) Compare to Q3 F_1.



(c) Compare to average accuracy.



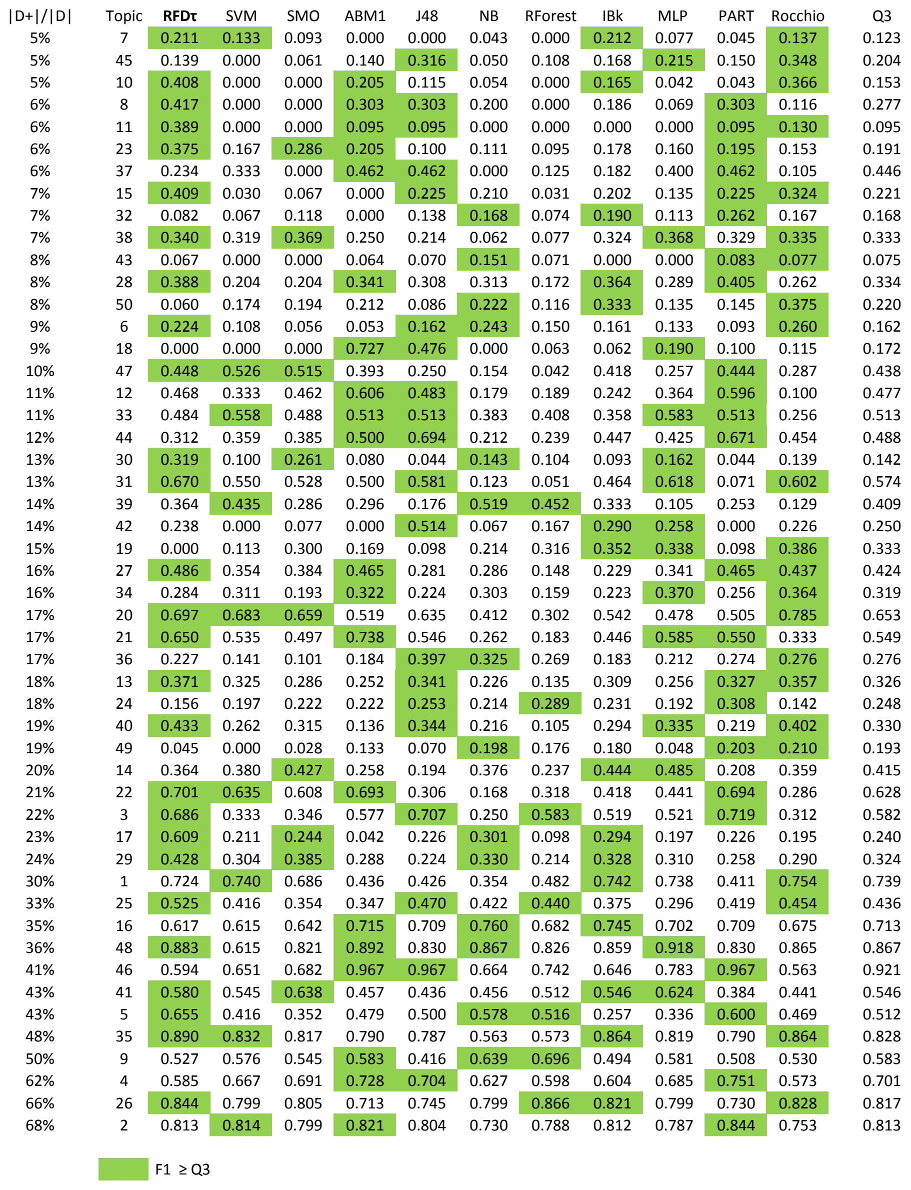
(d) Compare to Q3 accuracy.

Figure 6.17: RFD$_\tau$ v.s. baseline models performance over training set imbalance rate.

Similar Trends in Training and Testing Document Weight The proposed model used a training dataset, especially minimum weight of positive document and maximum weight of negative documents. Figure 6.19 shows that the trends in the minimum value for the positive training and testing document were similar.

Decision Boundary Setting The results showed that the performance of the initial decision boundary setting (τ_{low}), which is used for basic calculation was better than the other alternatives (Table 6.24 and Table 6.25).

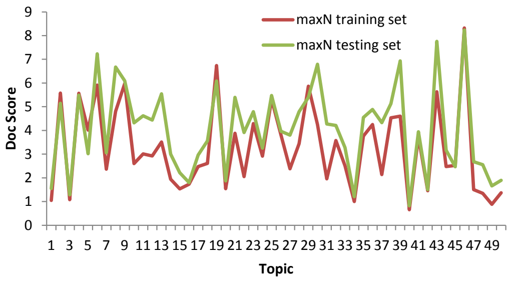
Influence of Outlier Removal The results showed that the removal of the outlier in the training dataset (for a large set) for large $|D^+|$ and $|D^-|$ increased the classification performance (Table 6.26 and Table 6.27). On the contrary, if $|D^+|$ and $|D^-|$ were small, outlier removal hurt the performance.



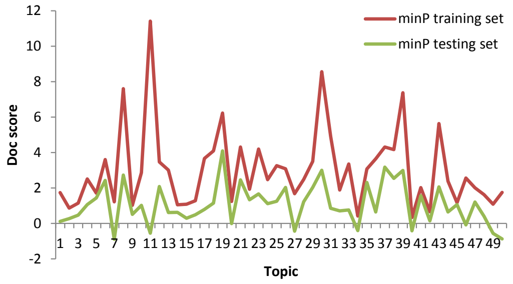
|D+|/|D|	Topic	**RFDτ**	SVM	SMO	ABM1	J48	NB	RForest	IBk	MLP	PART	Rocchio	Q3
5%	7	🟩 0.211	🟩 0.133	0.093	0.000	0.000	0.043	0.000	🟩 0.212	0.077	0.045	🟩 0.137	0.123
5%	45	0.139	0.000	0.061	0.140	🟩 0.316	0.050	0.108	0.168	🟩 0.215	0.150	🟩 0.348	0.204
5%	10	🟩 0.408	0.000	0.000	🟩 0.205	0.115	0.054	0.000	🟩 0.165	0.042	0.043	🟩 0.366	0.153
6%	8	🟩 0.417	0.000	0.000	🟩 0.303	🟩 0.303	0.200	0.000	0.186	0.069	🟩 0.303	0.116	0.277
6%	11	🟩 0.389	0.000	0.000	🟩 0.095	🟩 0.095	0.000	0.000	0.000	0.000	🟩 0.095	🟩 0.130	0.095
6%	23	🟩 0.375	0.167	🟩 0.286	🟩 0.205	0.100	0.111	0.095	0.178	0.160	🟩 0.195	0.153	0.191
6%	37	0.234	0.333	0.000	🟩 0.462	🟩 0.462	0.000	0.125	0.182	0.400	🟩 0.462	0.105	0.446
7%	15	🟩 0.409	0.030	0.067	0.000	🟩 0.225	0.210	0.031	0.202	0.135	🟩 0.225	🟩 0.324	0.221
7%	32	0.082	0.067	0.118	0.000	0.138	🟩 0.168	0.074	🟩 0.190	0.113	🟩 0.262	0.167	0.168
7%	38	🟩 0.340	0.319	🟩 0.369	0.250	0.214	0.062	0.077	0.324	🟩 0.368	0.329	🟩 0.335	0.333
8%	43	0.067	0.000	0.000	0.064	0.070	🟩 0.151	0.071	0.000	0.000	🟩 0.083	🟩 0.077	0.075
8%	28	🟩 0.388	0.204	0.204	🟩 0.341	0.308	0.313	0.172	🟩 0.364	0.289	🟩 0.405	0.262	0.334
8%	50	0.060	0.174	0.194	0.212	0.086	🟩 0.222	0.116	🟩 0.333	0.135	0.145	🟩 0.375	0.220
9%	6	🟩 0.224	0.108	0.056	0.053	🟩 0.162	🟩 0.243	0.150	0.161	0.133	0.093	🟩 0.260	0.162
9%	18	0.000	0.000	0.000	🟩 0.727	🟩 0.476	0.000	0.063	0.062	🟩 0.190	0.100	0.115	0.172
10%	47	🟩 0.448	🟩 0.526	🟩 0.515	0.393	0.250	0.154	0.042	0.418	0.257	🟩 0.444	0.287	0.438
11%	12	0.468	0.333	0.462	🟩 0.606	🟩 0.483	0.179	0.189	0.242	0.364	🟩 0.596	0.100	0.477
11%	33	0.484	🟩 0.558	0.488	🟩 0.513	🟩 0.513	0.383	0.408	0.358	🟩 0.583	🟩 0.513	0.256	0.513
12%	44	0.312	0.359	0.385	🟩 0.500	🟩 0.694	0.212	0.239	0.447	0.425	🟩 0.671	0.454	0.488
13%	30	🟩 0.319	0.100	🟩 0.261	0.080	0.044	🟩 0.143	0.104	0.093	🟩 0.162	0.044	0.139	0.142
13%	31	🟩 0.670	0.550	0.528	0.500	🟩 0.581	0.123	0.051	0.464	🟩 0.618	0.071	🟩 0.602	0.574
14%	39	0.364	🟩 0.435	0.286	0.296	0.176	🟩 0.519	🟩 0.452	0.333	0.105	0.253	0.129	0.409
14%	42	0.238	0.000	0.077	0.000	🟩 0.514	0.067	0.167	🟩 0.290	🟩 0.258	0.000	0.226	0.250
15%	19	0.000	0.113	0.300	0.169	0.098	0.214	0.316	🟩 0.352	🟩 0.338	0.098	🟩 0.386	0.333
16%	27	🟩 0.486	0.354	0.384	🟩 0.465	0.281	0.286	0.148	0.229	0.341	🟩 0.465	🟩 0.437	0.424
16%	34	0.284	0.311	0.193	🟩 0.322	0.224	0.303	0.159	0.223	🟩 0.370	0.256	🟩 0.364	0.319
17%	20	🟩 0.697	🟩 0.683	🟩 0.659	0.519	0.635	0.412	0.302	0.542	0.478	0.505	🟩 0.785	0.653
17%	21	🟩 0.650	0.535	0.497	🟩 0.738	0.546	0.262	0.183	0.446	🟩 0.585	🟩 0.550	0.333	0.549
17%	36	0.227	0.141	0.101	0.184	🟩 0.397	🟩 0.325	0.269	0.183	0.212	0.274	🟩 0.276	0.276
18%	13	🟩 0.371	0.325	0.286	0.252	🟩 0.341	0.226	0.135	0.309	0.256	🟩 0.327	🟩 0.357	0.326
18%	24	0.156	0.197	0.222	0.222	🟩 0.253	0.214	🟩 0.289	0.231	0.192	🟩 0.308	0.142	0.248
19%	40	🟩 0.433	0.262	0.315	0.136	🟩 0.344	0.216	0.105	0.294	🟩 0.335	0.219	🟩 0.402	0.330
19%	49	0.045	0.000	0.028	0.133	0.070	🟩 0.198	0.176	0.180	0.048	🟩 0.203	🟩 0.210	0.193
20%	14	0.364	0.380	🟩 0.427	0.258	0.194	0.376	0.237	🟩 0.444	🟩 0.485	0.208	0.359	0.415
21%	22	🟩 0.701	🟩 0.635	0.608	🟩 0.693	0.306	0.168	0.318	0.418	0.441	🟩 0.694	0.286	0.628
22%	3	🟩 0.686	0.333	0.346	0.577	🟩 0.707	0.250	🟩 0.583	0.519	0.521	🟩 0.719	0.312	0.582
23%	17	🟩 0.609	0.211	🟩 0.244	0.042	0.226	🟩 0.301	0.098	🟩 0.294	0.197	0.226	0.195	0.240
24%	29	🟩 0.428	0.304	🟩 0.385	0.288	0.224	🟩 0.330	0.214	🟩 0.328	0.310	0.258	0.290	0.324
30%	1	0.724	🟩 0.740	0.686	0.436	0.426	0.354	0.482	🟩 0.742	0.738	0.411	🟩 0.754	0.739
33%	25	🟩 0.525	0.416	0.354	0.347	🟩 0.470	0.422	🟩 0.440	0.375	0.296	0.419	🟩 0.454	0.436
35%	16	0.617	0.615	0.642	🟩 0.715	0.709	🟩 0.760	0.682	🟩 0.745	0.702	0.709	0.675	0.713
36%	48	🟩 0.883	0.615	0.821	🟩 0.892	0.830	🟩 0.867	0.826	0.859	🟩 0.918	0.830	0.865	0.867
41%	46	0.594	0.651	0.682	🟩 0.967	🟩 0.967	0.664	0.742	0.646	0.783	🟩 0.967	0.563	0.921
43%	41	🟩 0.580	0.545	🟩 0.638	0.457	0.436	0.456	0.512	🟩 0.546	🟩 0.624	0.384	0.441	0.546
43%	5	🟩 0.655	0.416	0.352	0.479	0.500	🟩 0.578	🟩 0.516	0.257	0.336	🟩 0.600	0.469	0.512
48%	35	🟩 0.890	🟩 0.832	0.817	0.790	0.787	0.563	0.573	🟩 0.864	0.819	0.790	🟩 0.864	0.828
50%	9	0.527	0.576	0.545	🟩 0.583	0.416	🟩 0.639	🟩 0.696	0.494	0.581	0.508	0.530	0.583
62%	4	0.585	0.667	0.691	🟩 0.728	🟩 0.704	0.627	0.598	0.604	0.685	🟩 0.751	0.573	0.701
66%	26	🟩 0.844	0.799	0.805	0.713	0.745	0.799	🟩 0.866	🟩 0.821	0.799	0.730	🟩 0.828	0.817
68%	2	0.813	🟩 0.814	0.799	🟩 0.821	0.804	0.730	0.788	0.812	0.787	🟩 0.844	0.753	0.813

🟩 F1 ≥ Q3

Figure 6.18: Visualisation of comparison of F_1 RFD_τ vs. baseline models sorted by training set imbalance rate. Shaded numbers means the same or higher than Q3.



(a) Max weight of neg documents.



(b) Min weight of pos documents.

Figure 6.19: Similar trend of training and testing document weight.

Table 6.24: Decision boundary setting.

Variant	Description
τ_{low}	$\tau = \tau_{low}$
τ_{α}	if $\tau_P > \tau_N$ then $\tau = \tau_N$
	else $\tau = \tau_P - \alpha$
	where α is average boundary distances.
τ_{β}	use harmonic mean of document's score from positive training and negative training set
	$\tau = (1+\beta^2)\frac{\tau_N \times \tau_P}{\beta^2 \tau_N + \tau_P}$
τ_{Pave}	$\tau = \frac{\sum_{d \in D^+} score(d)}{\lvert D^+ \rvert}$

Table 6.25: RFD$_{\tau}$ with different threshold settings.

Variant	Macroaverage		Microaverage	
	F_1	Acc	F_1	Acc
τ_{low}	**0.426**	**0.682**	**0.527**	**0.701**
τ_{α}	0.385	0.623	0.410	0.620
$\tau_{\beta=2}$	0.292	0.602	0.435	0.632
$\tau_{\beta=3}$	0.362	0.641	0.504	0.677
$\tau_{\beta=5}$	0.420	0.672	0.522	0.692
τ_P	0.362	0.636	0.490	0.634
τ_N	0.306	0.633	0.435	0.634
τ_{Pave}	0.121	0.533	0.209	0.542

Table 6.26: Performance of topics with outliers and suspected outliers in D^+, after the outliers have been removed.

	Difference from RFD$_{\tau'}$		
Topic	F_1	Acc	$\|D^+\|$
102	0.025	0.047	135
104	0.041	0.080	120
106	-0.138	-0.129	4
114	0.054	-0.008	5
119	0.000	0.000	4
128	-0.001	-0.091	4
134	-0.236	-0.067	5
135	-0.067	0.006	14
146	-0.104	0.023	13

Table 6.27: Performance of topics with outliers or suspected outliers in D^-, after the outliers have been removed.

	Difference from RFD$_{\tau'}$		
Topic	F_1	Acc	$\|D^-\|$
101	0.091	-0.093	16
103	0.039	0.074	50
108	0.266	0.167	50
111	0.051	0.033	49
148	0.002	0.003	21

6.6 Evaluation of Classifier Combination

In this study, the classifier combination approach was used to improve the performance of the RFD$_\tau$ classifier. This section presents the evaluation of the proposed classifier combination approach for RFD$_\tau$. The proposed classifier combination is called RFD$_{CC}$.

6.6.1 Evaluation Procedures

Figure 5.4 shows the classifier combination process. Each process of classification was conducted as in Figure 6.3. As shown in Figure 5.4, a testing document will be predicted as positive if it is predicted as positive in both classifiers; otherwise, it will be predicted as negative.

6.6.2 Results

The results, as summarised in Table 6.28, showed that RFD$_\tau$-Rocchio had the best performance among combination approaches for RFD$_\tau$. This RFD$_\tau$-Rocchio combination increased the performance of the RFD$_\tau$ (see Table 6.29). Table 6.30 presents a comparison of the RFD$_\tau$-Rocchio with the other classifier combination models. As shown in the table, shows the RFD$_\tau$-Rocchio combination had the best performance.

Table 6.28: Classifier combination models of RFD$_{\tau}$.

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_{\tau}$-Rocchio	**0.441**	**0.697**	**0.569**	**0.717**
RFD$_{\tau}$-SVM	0.192	0.567	0.462	0.616
RFD$_{\tau}$-SMO	0.325	0.616	0.542	0.668
RFD$_{\tau}$-ABM1	0.329	0.617	0.550	0.668
RFD$_{\tau}$-J48	0.317	0.628	0.516	0.613
RFD$_{\tau}$-NB	0.256	0.585	0.469	0.627
RFD$_{\tau}$-RForest	0.264	0.589	0.500	0.638
RFD$_{\tau}$-IBk	0.276	0.610	0.508	0.660
RFD$_{\tau}$-MLP	0.339	0.630	0.550	0.679
RFD$_{\tau}$-PART	0.342	0.620	0.527	0.662

Table 6.29: Comparison of RFD$_{\tau}$-Rocchio and RFD$_{\tau}$.

Model	Macro-average		Micro-average	
	F_1^M	Acc^M	F_1^μ	Acc^μ
RFD$_{CC}$	**0.441**	**0.697**	**0.569**	**0.717**
RFD$\mathcal{T}$	0.428	0.688	0.537	0.711

Table 6.30: Comparison of RFD$_\tau$-Rocchio and other classifier combination models with TF$\times$IDF term weight (best performance).

Model	Macro-average		Micro-average	
	F_1	Acc	F_1	Acc
RFD$_{CC}$	**0.441**	**0.697**	**0.569**	**0.717**
SVM-SMO	0.189	0.563	0.458	0.607
SVM-ABoost	0.186	0.561	0.458	0.605
SVM-J48	0.174	0.555	0.423	0.596
SVM-NB	0.168	0.556	0.425	0.598
SVM-RForest	0.171	0.556	0.432	0.597
SVM-Ibk	0.157	0.564	0.433	0.612
SVM-MLP	0.190	0.564	0.461	0.608
SVM-PART	0.176	0.556	0.428	0.597
SVM-Rocchio	0.192	0.564	0.462	0.609
SMO-ABoost	0.283	0.589	0.531	0.640
SMO-J48	0.267	0.586	0.493	0.630
SMO-NB	0.238	0.577	0.453	0.615
SMO-RForest	0.234	0.572	0.484	0.618
SMO-Ibk	0.259	0.596	0.505	0.649
SMO-MLP	0.329	0.614	0.544	0.660
SMO-PART	0.275	0.587	0.492	0.630
SMO-Rocchio	0.350	0.623	0.549	0.672
ABoost-J48	0.298	0.595	0.504	0.629
ABoost-NB	0.223	0.571	0.450	0.609
ABoost-RForest	0.237	0.572	0.489	0.613
ABoost-Ibk	0.235	0.589	0.507	0.646
ABoost-MLP	0.288	0.598	0.538	0.648
ABoost-PART	0.301	0.594	0.501	0.627
ABoost-Rocchio	0.354	0.620	0.566	0.666
J48-NB	0.215	0.568	0.435	0.606
J48-RForest	0.230	0.570	0.455	0.606
J48-Ibk	0.218	0.581	0.454	0.624
J48-MLP	0.275	0.593	0.497	0.634
J48-PART	0.331	0.614	0.509	0.636
J48-Rocchio	0.342	0.623	0.519	0.654
NB-RForest	0.208	0.566	0.432	0.601
NB-Ibk	0.188	0.569	0.406	0.606
NB-MLP	0.238	0.578	0.452	0.616

Continued on next page

Table 6.31: Recall-oriented Rocchio.

Model	Macro-average		Micro-average	
	Rec	$Prec$	Rec	$Prec$
Classifier 1: Rocchio	**0.825**	0.268	**0.850**	0.281
Classifier 2: RFD$_\tau$	0.589	0.385	0.686	0.441

Table 6.30 – continued from previous page

Model	Macro-average		Micro-average	
	F_1	Acc	F_1	Acc
NB-PART	0.231	0.573	0.450	0.613
NB-Rocchio	0.269	0.588	0.460	0.625
RForest-Ibk	0.205	0.577	0.464	0.625
RForest-MLP	0.244	0.580	0.493	0.624
RForest-PART	0.233	0.569	0.458	0.606
RForest-Rocchio	0.272	0.588	0.513	0.634
Ibk-MLP	0.263	0.610	0.516	0.664
Ibk-PART	0.236	0.595	0.467	0.639
Ibk-Rocchio	0.311	0.625	0.518	0.674
MLP-PART	0.286	0.603	0.505	0.644
MLP-Rocchio	0.362	0.641	0.559	0.689
PART-Rocchio	0.378	0.636	0.537	0.668

6.6.3 Discussion

The results of the evaluation (Figure 6.20, Table 6.31, and Table 6.32) indicated that the Rocchio classifier is a recall-oriented.

The results in Table 6.33 (presented in more detail in Table 6.34) showed that the strong recall oriented Rocchio classifier changed positive predictions (TP and FP) into negative prediction (TN and FN). With a greater increase in TN than FN, makes the final performance of the RFD$_\tau$-Rocchio increased.

Table 6.32: Experiment results (including their recall and precision) with TF$\times$IDF term weight for baseline models (best performance).

Model	Macro-average				Micro-average			
	R^M	P^M	F_1^M	Acc^M	R^μ	P^μ	F_1^μ	Acc^μ
SVM linear	0.301	0.440	0.337	0.611	0.478	0.643	0.549	0.656
SVM poly	0.080	0.036	0.049	0.500	0.143	0.455	0.218	0.500
SVM radial	0.099	0.112	0.095	0.517	0.226	0.630	0.332	0.528
SVM sigmoid	0.099	0.058	0.069	0.509	0.225	0.555	0.320	0.515
SMO norm-poly	0.186	0.359	0.199	0.570	0.373	0.760	0.500	0.623
SMO poly	0.309	0.456	0.345	0.616	0.488	0.650	0.557	0.661
SMO Puk	0.123	0.268	0.139	0.544	0.244	0.751	0.368	0.574
ABM1 base d. stump	0.320	0.461	0.350	0.623	0.499	0.658	0.568	0.658
ABM1 base J48	0.356	0.446	0.377	0.628	0.480	0.607	0.536	0.656
J48 pruned	0.374	0.382	0.354	0.626	0.478	0.528	0.502	0.645
J48 unpruned	0.395	0.402	0.379	0.631	0.526	0.506	0.516	0.648
BayesNet	0.269	0.378	0.285	0.605	0.392	0.673	0.495	0.627
NB normal distr	0.321	0.369	0.303	0.590	0.430	0.406	0.418	0.607
NB kernel density	0.190	0.333	0.203	0.564	0.366	0.626	0.462	0.593
Random Forest	0.255	0.373	0.280	0.588	0.389	0.554	0.457	0.615
IBk k=1	0.322	0.363	0.326	0.600	0.494	0.532	0.512	0.647
IBk k=2	0.446	0.331	0.363	0.619	0.621	0.437	0.513	0.647
MLP hidden=1	0.359	0.410	0.359	0.627	0.517	0.568	0.541	0.664
MLP hidden=a	0.368	0.397	0.365	0.628	0.540	0.557	0.549	0.669
PART	0.401	0.401	0.376	0.635	0.505	0.532	0.518	0.655
Rocchio	0.840	0.267	0.362	0.654	0.845	0.265	0.403	0.675

Table 6.33: More negative prediction in RFD$_\tau$-Rocchio than in RFD$_\tau$

Model	TP	FP	TN	FN	R	P	F$_1$
RFD$_\tau$-Rocchio	2169	1968	13449	1315	0.623	0.524	0.569
RFD$_\tau$	2389	3026	12391	1095	0.686	0.441	0.537
	-220	-1058	1058	220			

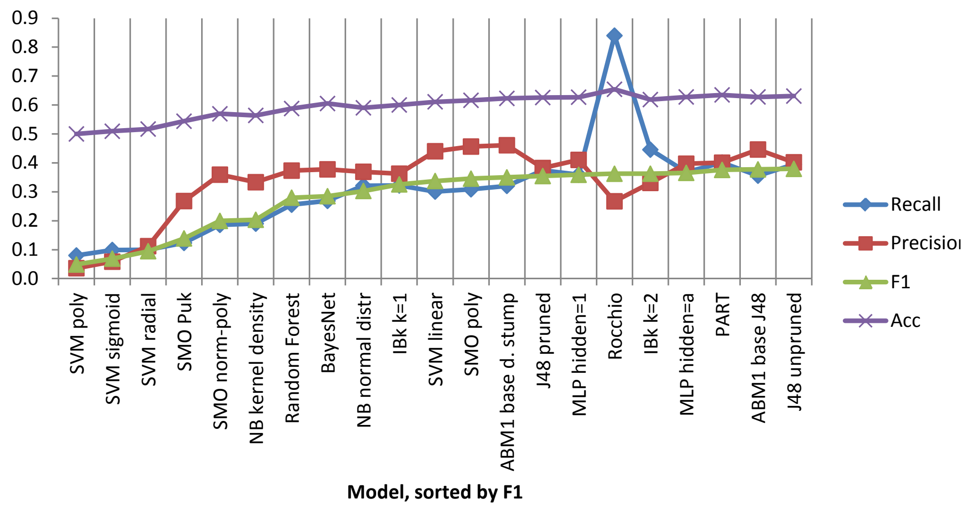


Figure 6.20: Experiment results (including their recall and precision) in macro-average with TF×IDF term weight for baseline models (best performance), sorted by F_1.

Table 6.34: RFD_τ-Rocchio in combination of Rocchio and RFD_τ.

		Classifier 1: Rocchio			
		TP	FP	TN	FN
Classifier 2: RFD_τ	TP	2169 (TP)			**220 (FN)**
	FP		1968 (FP)	**1058 (TN)**	
	TN		5624 (TN)	6767 (TN)	
	FN	791 (FN)			304 (FN)

Chapter 7

Conclusion

An important issue in text classification is the optimal identification of the boundary between classes. In a classifier, after features are selected, the boundary is still unclear with mixed positive and negative documents.

The main contribution of this research is an effective decision boundary setting approach. In this thesis, an effective training-based boundary decision setting method has been proposed in order to address that challenging issue to produce an effective text classifier, called RFD$_\tau$. To set the decision boundary, the initially boundary region is identified. Then, an initial decision boundary is set based on this region. After that, the initial boundary is adjusted or tuned. Classifier effectiveness can also be increased by decision swapping on topics with uncertain boundary based on specific terms. The present research included an intensive evaluation using comprehensive baseline models, several number of selected features set, two term weighting schemes, and macro & micro average evaluation metrics of F$_1$ and accuracy. The experiments demonstrated that the proposed decision boundary setting can produce a classifier that significantly outperforms existing classifiers, including state of the art classifiers.

Another contribution of this thesis is a classifier combination to boost classifier effectiveness. In this thesis, Rocchio which has a very high recall and precision weaker than RFD$_\tau$ but still moderate is used to combine with RFD$_\tau$ to boost classifier effectiveness.

This thesis used a binary dataset for the evaluation. As described in Chapter 1, multi-class and multi-label classifiers can be built based on binary classifiers. A potential work for future research is how to extend the proposed classifiers for multi-class and multi-label data, including for a big data. The proposed classifiers are evaluated using Reuters collections, evaluating them on different types of corpora is important in the future. In this thesis the proposed decision boundary setting method was applied to a pattern-based feature selection scheme. It would be important to investigate the use of other types of effective feature selection schemes in future work. In this thesis, the classifier combination boosted the performance of the proposed classifier. How to increase existing classifiers with recall oriented or precision oriented classifier is an interesting challenge to be pursued in the future.

Appendix A

Performance Difference: RFD$_\mathcal{T}$ vs. Baseline Models in All Topics

Figure A.1 shows the F$_1$ difference of the baseline models from RFD$_\tau$ sorted by F$_1$ of RFD$_\tau$. The baseline models are chosen from the best in each classification type. This figure shows that the better F$_1$ of RFD$_\tau$, the wider the gap.

Figure A.2 is similar to Figure A.1 for RFD$_\tau$ max or ideal. This figure indicates that an ideal version of RFD$_\tau$ outperforms almost all baseline models in almost all topics.

All models use the TFxIDF term weighting scheme.

Topic	F1 RFDτ	|D+|/|D|	SVM	SMO	ABM1	J48	NB	Random Forest	IBk	MLP	PART	Rocchio	Ave
18	0	9%	0	0	-1.455	-0.952	0	-0.125	-0.123	-0.381	-0.200	-0.229	-0.35
19	0	15%	-0.226	-0.600	-0.339	-0.197	-0.429	-0.632	-0.704	-0.677	-0.197	-0.773	-0.48
43	0.056	8%	0.100	0.100	-0.015	-0.026	-0.172	-0.029	0.100	0.100	-0.050	-0.038	0.007
50	0.063	8%	-0.198	-0.234	-0.265	-0.041	-0.284	-0.095	-0.481	-0.129	-0.146	-0.556	-0.24
49	0.065	19%	0.114	0.065	-0.122	-0.010	-0.236	-0.197	-0.205	0.030	-0.245	-0.257	-0.11
32	0.119	7%	0.085	0.003	0.193	-0.030	-0.078	0.073	-0.115	0.011	-0.231	-0.078	-0.02
45	0.135	5%	0.212	0.116	-0.008	-0.286	0.133	0.042	-0.053	-0.127	-0.024	-0.336	-0.03
24	0.164	18%	-0.049	-0.087	-0.087	-0.134	-0.075	-0.188	-0.100	-0.042	-0.216	0.034	-0.09
7	0.188	5%	0.079	0.137	0.273	0.273	0.211	0.273	-0.036	0.161	0.207	0.073	0.165
6	0.214	9%	0.148	0.221	0.226	0.072	-0.042	0.089	0.073	0.112	0.169	-0.064	0.1
37	0.247	6%	-0.116	0.330	-0.288	-0.288	0.330	0.163	0.087	-0.206	-0.288	0.189	-0.01
42	0.253	14%	0.336	0.234	0.336	-0.347	0.248	0.115	-0.049	-0.007	0.336	0.037	0.124
36	0.268	17%	0.165	0.217	0.109	-0.168	-0.074	-0.002	0.110	0.072	-0.008	-0.012	0.041
38	0.29	7%	-0.036	-0.100	0.051	0.096	0.289	0.270	-0.042	-0.098	-0.048	-0.056	0.033
34	0.3	16%	-0.015	0.134	-0.028	0.094	-0.004	0.176	0.096	-0.088	0.055	-0.081	0.034
30	0.333	13%	0.280	0.087	0.304	0.347	0.229	0.275	0.288	0.205	0.347	0.234	0.26
10	0.339	5%	0.404	0.404	0.160	0.267	0.340	0.404	0.208	0.355	0.352	-0.032	0.286
39	0.35	14%	-0.100	0.076	0.063	0.204	-0.198	-0.120	0.020	0.288	0.114	0.261	0.061
11	0.357	6%	0.417	0.417	0.306	0.306	0.417	0.417	0.417	0.417	0.306	0.265	0.368
14	0.364	20%	-0.019	-0.074	0.122	0.196	-0.014	0.147	-0.094	-0.140	0.180	0.006	0.031
15	0.368	7%	0.390	0.347	0.424	0.165	0.182	0.389	0.191	0.269	0.165	0.051	0.257
13	0.373	18%	0.054	0.100	0.138	0.036	0.168	0.273	0.072	0.134	0.053	0.017	0.104
12	0.379	11%	0.052	-0.094	-0.258	-0.118	0.227	0.217	0.156	0.018	-0.246	0.318	0.027
44	0.395	12%	0.041	0.012	-0.117	-0.334	0.205	0.175	-0.058	-0.033	-0.308	-0.065	-0.05
28	0.397	8%	0.215	0.215	0.062	0.100	0.094	0.250	0.037	0.120	-0.009	0.151	0.124
29	0.426	24%	0.131	0.044	0.148	0.217	0.103	0.228	0.105	0.125	0.181	0.147	0.143
47	0.452	10%	-0.078	-0.066	0.062	0.212	0.313	0.431	0.036	0.205	0.008	0.174	0.13
27	0.495	16%	0.141	0.112	0.030	0.216	0.211	0.349	0.268	0.155	0.030	0.058	0.157
40	0.497	19%	0.236	0.183	0.362	0.153	0.282	0.393	0.203	0.162	0.279	0.095	0.235
33	0.5	11%	-0.058	0.012	-0.013	-0.013	0.117	0.092	0.142	-0.083	-0.013	0.244	0.043
9	0.523	50%	-0.051	-0.022	-0.059	0.105	-0.113	-0.169	0.029	-0.057	0.015	-0.007	-0.03
25	0.527	33%	0.108	0.169	0.175	0.056	0.102	0.085	0.149	0.225	0.106	0.072	0.125
23	0.529	6%	0.352	0.237	0.315	0.417	0.406	0.422	0.342	0.359	0.325	0.366	0.354
8	0.571	6%	0.533	0.533	0.251	0.251	0.347	0.533	0.360	0.469	0.251	0.425	0.395
4	0.587	62%	-0.073	-0.096	-0.130	-0.107	-0.037	-0.010	-0.015	-0.090	-0.151	0.013	-0.07
46	0.594	41%	-0.052	-0.080	-0.341	-0.341	-0.063	-0.135	-0.047	-0.172	-0.341	0.029	-0.15
41	0.595	43%	0.045	-0.039	0.126	0.145	0.127	0.076	0.045	-0.026	0.193	0.141	0.083
3	0.6	22%	0.242	0.231	0.021	-0.097	0.318	0.015	0.074	0.072	-0.108	0.262	0.103
17	0.609	23%	0.359	0.329	0.511	0.345	0.277	0.461	0.284	0.371	0.345	0.373	0.366
16	0.628	35%	0.011	-0.012	-0.077	-0.071	-0.117	-0.048	-0.103	-0.065	-0.071	-0.041	-0.06
5	0.661	43%	0.211	0.267	0.157	0.139	0.072	0.125	0.348	0.280	0.053	0.166	0.182
31	0.674	13%	0.106	0.124	0.149	0.079	0.469	0.531	0.179	0.048	0.513	0.062	0.226
1	0.683	30%	-0.048	-0.002	0.209	0.218	0.278	0.170	-0.049	-0.047	0.230	-0.060	0.09
20	0.685	17%	0.002	0.021	0.140	0.042	0.230	0.323	0.121	0.175	0.152	-0.084	0.112
21	0.685	17%	0.127	0.159	-0.045	0.117	0.357	0.424	0.202	0.084	0.114	0.297	0.184
22	0.774	21%	0.109	0.130	0.063	0.367	0.475	0.358	0.279	0.261	0.063	0.383	0.249
2	0.814	68%	0.000	0.012	-0.005	0.007	0.064	0.020	0.002	0.020	-0.023	0.046	0.014
26	0.885	66%	0.062	0.058	0.124	0.101	0.062	0.013	0.047	0.062	0.112	0.041	0.068
48	0.892	36%	0.199	0.051	0.000	0.045	0.018	0.047	0.024	-0.018	0.045	0.019	0.043
35	0.893	48%	0.044	0.054	0.074	0.076	0.237	0.229	0.021	0.053	0.074	0.020	0.088

Low 🟥🟩 High

Figure A.1: Performance difference RFD$_\tau$ vs baseline models.

Topic	F1 RFDτ Ideal	|D+|/|D|	SVM	SMO	ABM1	J48	NB	Random Forest	IBk	MLP	PART	Rocchio	Ave
18	0.208	9%	0.293	0.293	-0.735	-0.380	0.293	0.205	0.206	0.024	0.152	0.131	0.048
19	0.557	15%	0.420	0.243	0.366	0.434	0.324	0.228	0.194	0.207	0.434	0.161	0.301
43	0.184	8%	0.269	0.269	0.175	0.167	0.048	0.164	0.269	0.269	0.147	0.157	0.193
50	0.377	8%	0.232	0.209	0.189	0.332	0.177	0.298	0.050	0.276	0.265	0.002	0.203
49	0.247	19%	0.330	0.293	0.152	0.236	0.065	0.095	0.089	0.266	0.059	0.049	0.163
32	0.171	7%	0.156	0.080	0.255	0.050	0.005	0.145	-0.028	0.088	-0.135	0.006	0.062
45	0.157	5%	0.239	0.146	0.026	-0.242	0.162	0.074	-0.018	-0.089	0.010	-0.291	0.002
24	0.259	18%	0.082	0.048	0.048	0.008	0.059	-0.040	0.037	0.088	-0.064	0.154	0.042
7	0.4	5%	0.296	0.341	0.444	0.444	0.397	0.444	0.209	0.359	0.394	0.292	0.362
6	0.252	9%	0.192	0.261	0.265	0.120	0.012	0.136	0.121	0.158	0.212	-0.010	0.147
37	0.56	6%	0.214	0.528	0.093	0.093	0.528	0.410	0.357	0.151	0.093	0.429	0.29
42	0.368	14%	0.424	0.336	0.424	-0.168	0.347	0.232	0.090	0.127	0.424	0.165	0.24
36	0.367	17%	0.260	0.306	0.211	-0.035	0.048	0.112	0.211	0.178	0.107	0.104	0.15
38	0.364	7%	0.052	-0.006	0.132	0.173	0.349	0.332	0.046	-0.005	0.041	0.033	0.115
34	0.321	16%	0.012	0.157	-0.001	0.119	0.023	0.198	0.120	-0.059	0.080	-0.052	0.06
30	0.429	13%	0.354	0.181	0.375	0.414	0.308	0.350	0.361	0.287	0.414	0.312	0.336
10	0.508	5%	0.504	0.504	0.301	0.390	0.451	0.504	0.341	0.463	0.461	0.141	0.406
39	0.647	14%	0.185	0.315	0.306	0.410	0.112	0.170	0.274	0.472	0.344	0.452	0.304
11	0.357	6%	0.417	0.417	0.306	0.306	0.417	0.417	0.417	0.417	0.306	0.265	0.368
14	0.437	20%	0.061	0.011	0.191	0.259	0.065	0.214	-0.008	-0.051	0.245	0.084	0.107
15	0.416	7%	0.421	0.381	0.454	0.208	0.225	0.420	0.233	0.307	0.208	0.100	0.296
13	0.444	18%	0.126	0.168	0.204	0.109	0.231	0.328	0.143	0.200	0.125	0.092	0.173
12	0.513	11%	0.177	0.051	-0.092	0.030	0.329	0.320	0.267	0.147	-0.082	0.408	0.155
44	0.512	12%	0.151	0.126	0.012	-0.180	0.296	0.270	0.064	0.086	-0.157	0.058	0.073
28	0.404	8%	0.221	0.221	0.069	0.106	0.101	0.256	0.044	0.127	-0.001	0.157	0.13
29	0.448	24%	0.152	0.066	0.168	0.236	0.125	0.247	0.126	0.146	0.200	0.167	0.163
47	0.46	10%	-0.069	-0.058	0.069	0.219	0.319	0.436	0.044	0.211	0.016	0.180	0.137
27	0.481	16%	0.129	0.100	0.017	0.205	0.199	0.340	0.257	0.143	0.017	0.045	0.145
40	0.487	19%	0.228	0.174	0.355	0.145	0.274	0.387	0.195	0.154	0.272	0.086	0.227
33	0.538	11%	-0.019	0.049	0.025	0.025	0.150	0.125	0.174	-0.043	0.025	0.272	0.078
9	0.526	50%	-0.048	-0.019	-0.056	0.107	-0.109	-0.165	0.032	-0.053	0.018	-0.004	-0.03
25	0.54	33%	0.119	0.179	0.185	0.067	0.113	0.096	0.159	0.234	0.116	0.083	0.135
23	0.48	6%	0.320	0.198	0.280	0.388	0.376	0.393	0.308	0.327	0.291	0.334	0.321
8	0.533	6%	0.516	0.516	0.223	0.223	0.323	0.516	0.336	0.449	0.223	0.404	0.373
4	0.697	62%	0.025	0.005	-0.026	-0.005	0.059	0.083	0.078	0.010	-0.045	0.103	0.029
46	0.633	41%	-0.016	-0.043	-0.295	-0.295	-0.027	-0.096	-0.011	-0.132	-0.295	0.062	-0.11
41	0.598	43%	0.048	-0.036	0.128	0.148	0.130	0.078	0.047	-0.023	0.195	0.143	0.086
3	0.737	22%	0.326	0.316	0.129	0.024	0.394	0.124	0.177	0.175	0.014	0.344	0.202
17	0.667	23%	0.391	0.362	0.536	0.377	0.313	0.488	0.319	0.402	0.377	0.404	0.397
16	0.74	35%	0.100	0.079	0.020	0.025	-0.016	0.046	-0.004	0.031	0.025	0.053	0.036
5	0.656	43%	0.208	0.263	0.153	0.135	0.068	0.121	0.345	0.277	0.048	0.162	0.178
31	0.727	13%	0.144	0.162	0.185	0.119	0.492	0.551	0.215	0.089	0.534	0.102	0.259
1	0.834	30%	0.071	0.111	0.298	0.306	0.360	0.264	0.069	0.071	0.317	0.060	0.193
20	0.701	17%	0.015	0.034	0.152	0.054	0.240	0.332	0.132	0.186	0.163	-0.070	0.124
21	0.698	17%	0.136	0.168	-0.034	0.127	0.364	0.430	0.210	0.094	0.123	0.304	0.192
22	0.78	21%	0.114	0.134	0.068	0.370	0.478	0.361	0.283	0.265	0.067	0.386	0.253
2	0.82	68%	0.005	0.016	0.000	0.012	0.068	0.024	0.006	0.025	-0.018	0.051	0.019
26	0.912	66%	0.080	0.076	0.141	0.119	0.080	0.032	0.065	0.080	0.129	0.059	0.086
48	0.896	36%	0.201	0.053	0.003	0.047	0.020	0.050	0.026	-0.016	0.047	0.022	0.045
35	0.89	48%	0.042	0.052	0.072	0.074	0.235	0.228	0.019	0.051	0.072	0.018	0.086

Low 🟥🟩 High

Figure A.2: Performance difference ideal RFD+τ vs baseline models.

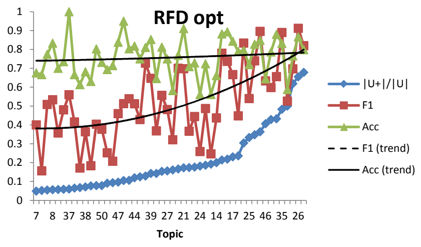
Appendix B

Performance Trend in Balance Rate of the Training Set

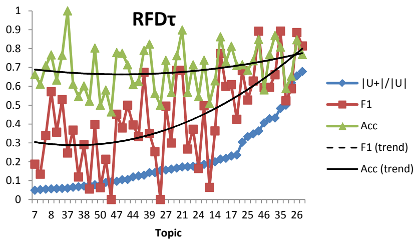
Figure B.1 and Figure B.2 show the performance trend of RFD$_\tau$ and baseline models, respectively, over the imbalance rate of the training set. All models use the TFxIDF term weighting scheme. The trendlines in this and other figures are generated by polynomial order two of the trend/regression type.

In the figures, the topics (horizontal axis) are sorted the by balance rate. The topic on the left side is the most imbalanced and the topic on the right side the most balanced.

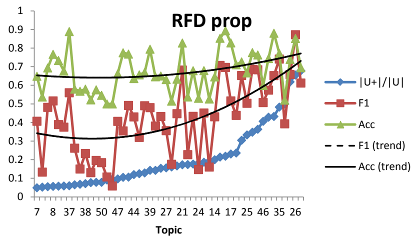
The figures show shows the same trend; that is, almost all the models perform better for lower the lower imbalance rate. However, we can see from the figures, as stated in Chapter 6 that the RFD$_\tau$ has better performance than the baseline models especially in regard to the high imbalance rate of the training set.



(a) RFD$_{\tau}$ optimal

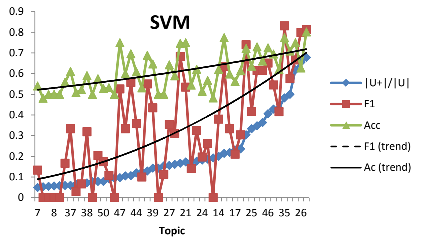


(b) RFD$_{\tau}$

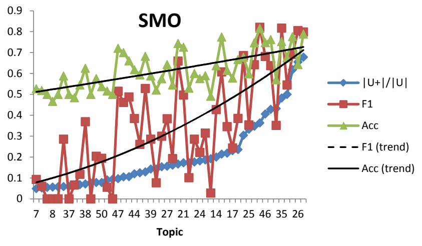


(c) RFD$_{prop}$

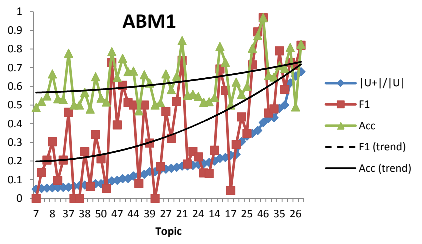
Figure B.1: Performance trend of RFD$_{\tau}$ over imbalance rate of training set.



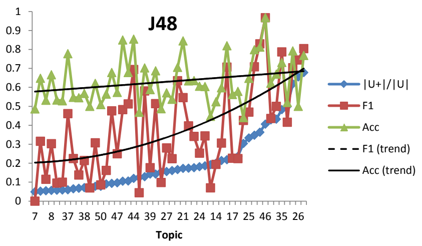
(a) SVM



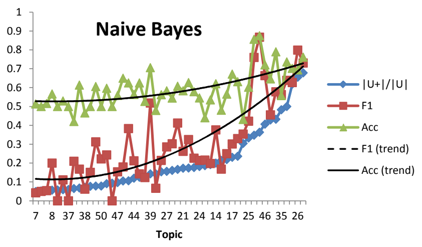
(b) SMO



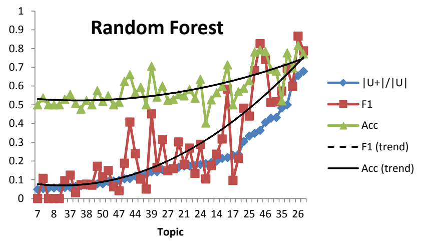
(c) AdaBoostM1



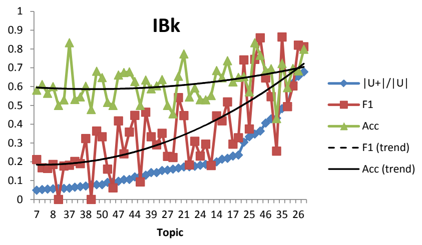
(d) J48



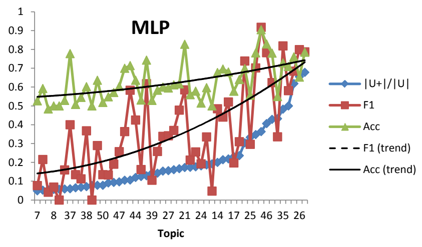
(e) Naive Bayes



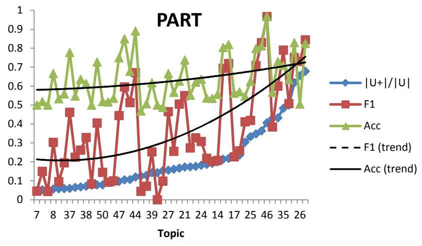
(f) Random Forest



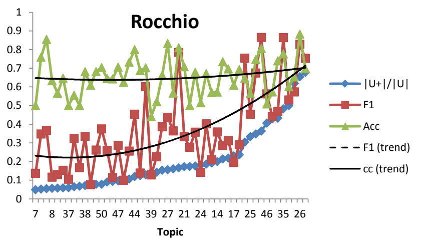
(g) IBk



(h) MLP



(i) PART



(j) Rocchio

Figure B.2: Performance trend of baseline models over imbalance rate of training set.

Appendix C

RFD$_{\mathcal{T}}$ Performance in Training Weight Distribution

The figures in this appendix show the performance of RFD$_{\mathcal{T}}$ at different points, from the minimal score to the maximum score of training documents (from $min(score(d \in D)$ to $max(score(d \in D))$. The left side of the horizontal axis states $min(score(d \in D)$ and the right side of the horizontal axis states $max(score(d \in D))$ of the topic.

In Figure C.2 to Figure C.6, the symbols under the horizontal axis denote the document positions based on their score. The top row is for training set D, and the bottom row is for testing set U. The symbol '+' on the right side indicates a positive document, and the symbol 'x' on the left side indicates a negative document.

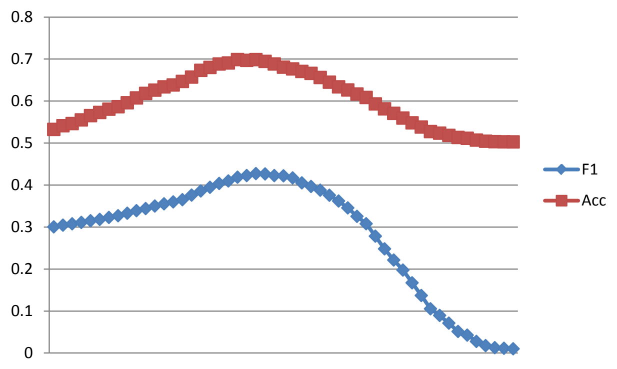
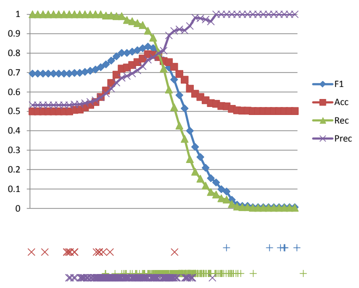
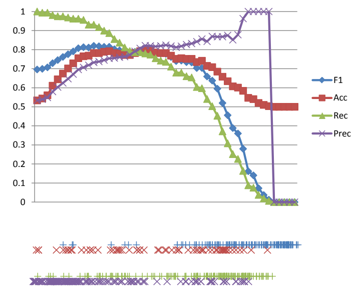


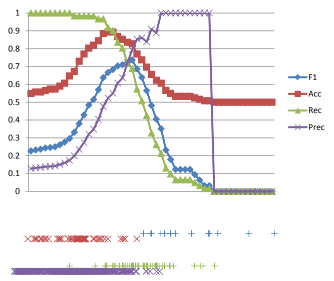
Figure C.1: RFD$_\tau$: average performance in training set weights distribution topic 1-50.



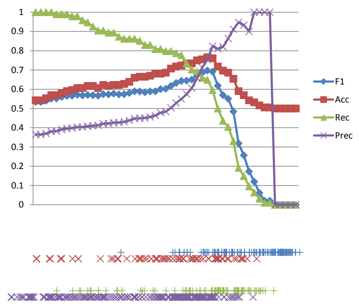
(a) Topic 1



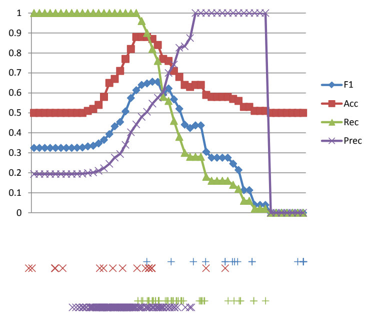
(b) Topic 2



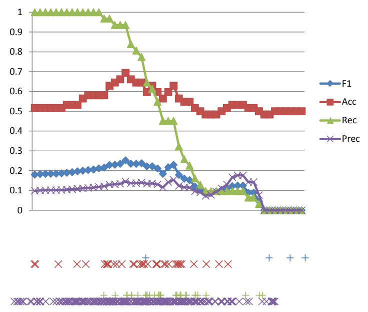
(c) Topic 3



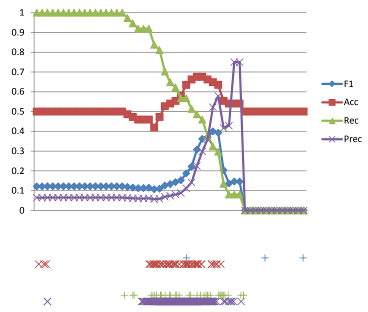
(d) Topic 4



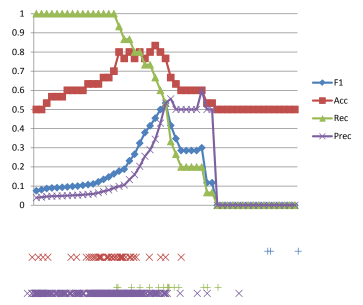
(e) Topic 5



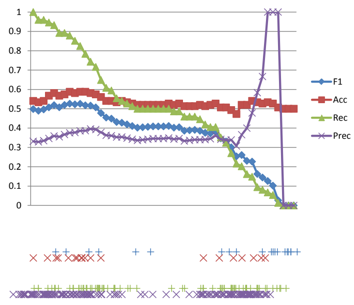
(f) Topic 6



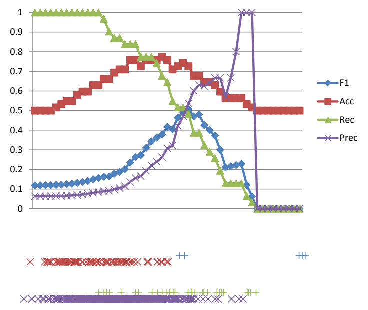
(g) Topic 7



(h) Topic 8

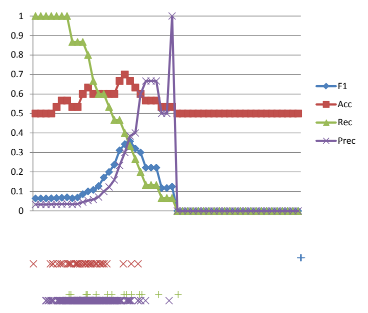


(i) Topic 9

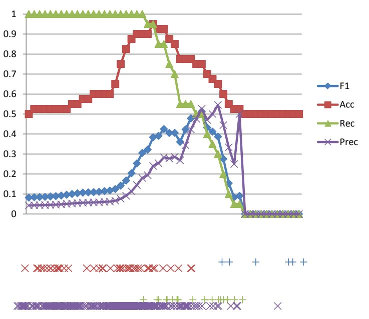


(j) Topic 10

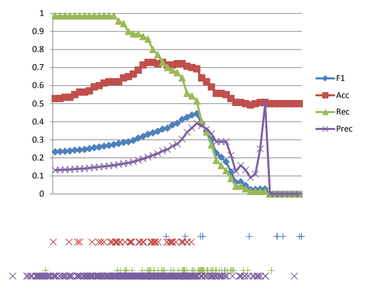
Figure C.2: RFD: performance over training set weights distribution topic 1-10.



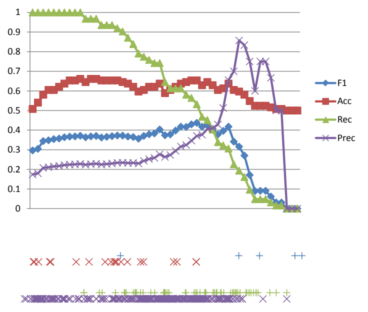
(a) Topic 11



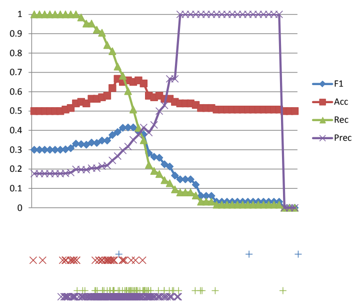
(b) Topic 12



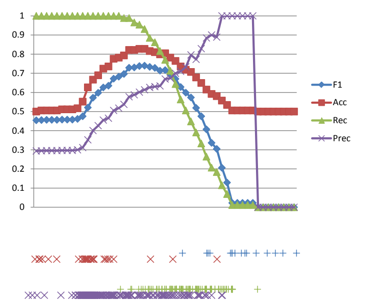
(c) Topic 13



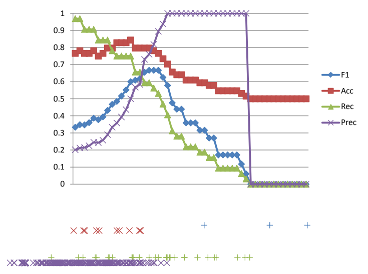
(d) Topic 14



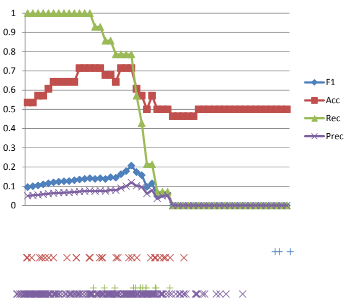
(e) Topic 15



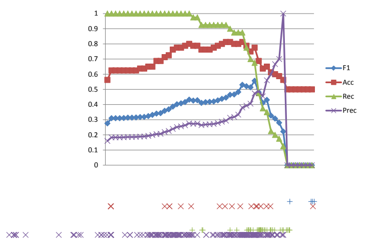
(f) Topic 16



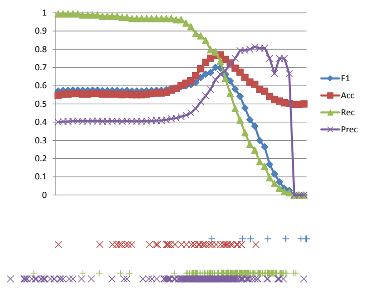
(g) Topic 17



(h) Topic 18

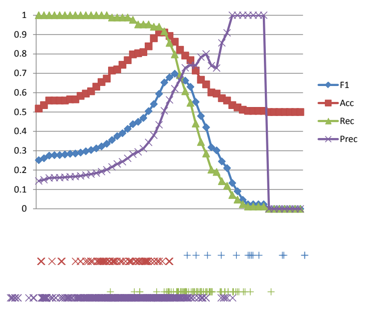


(i) Topic 19

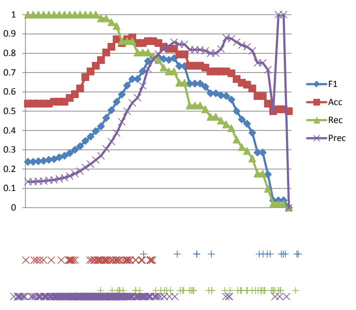


(j) Topic 20

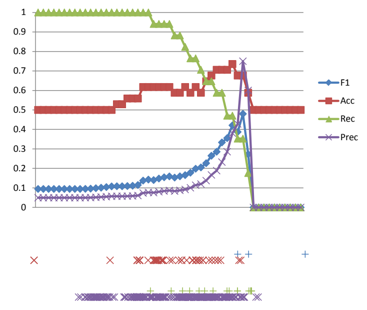
Figure C.3: RFD: performance in training set weights distribution topic 11-20.



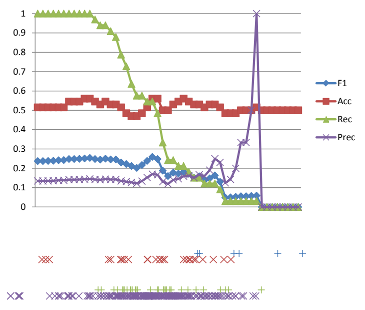
(a) Topic 21



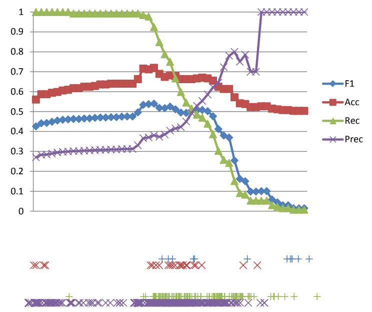
(b) Topic 22



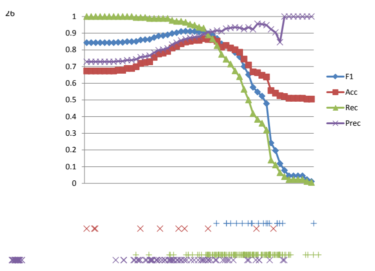
(c) Topic 23



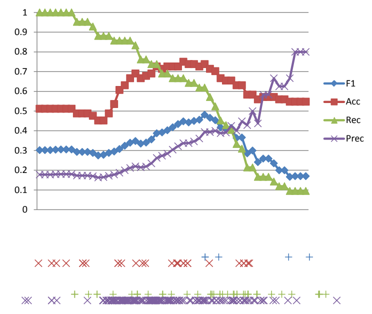
(d) Topic 24



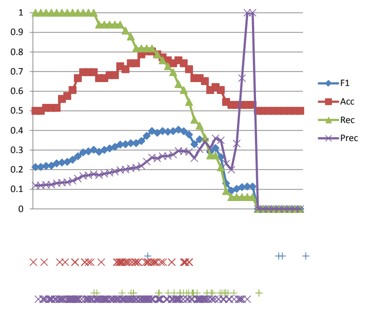
(e) Topic 25



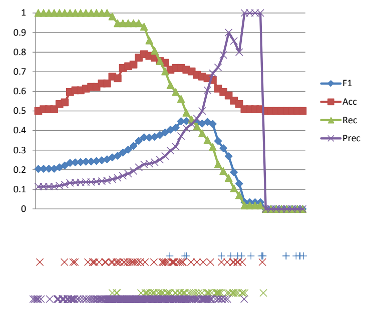
(f) Topic 26



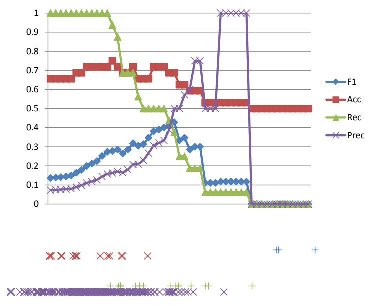
(g) Topic 27



(h) Topic 28

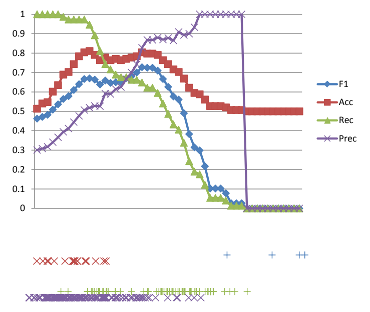


(i) Topic 29

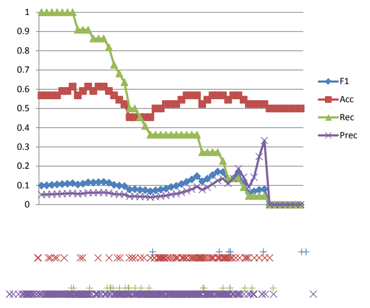


(j) Topic 30

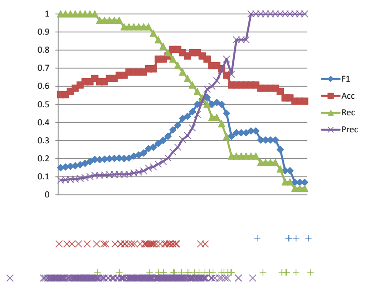
Figure C.4: RFD: performance in training set weights distribution topic 21-30.



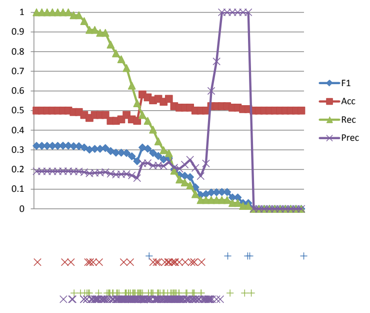
(a) Topic 31



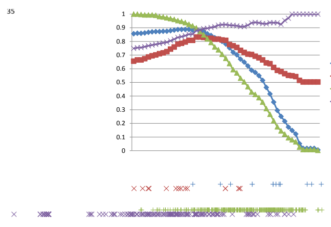
(b) Topic 32



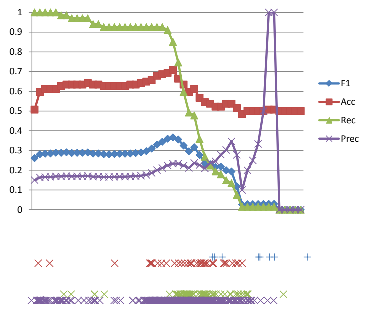
(c) Topic 33



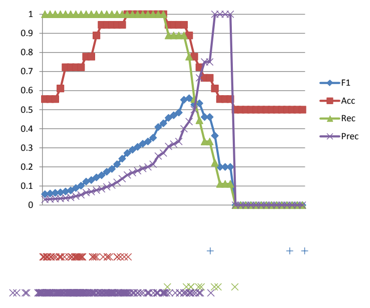
(d) Topic 34



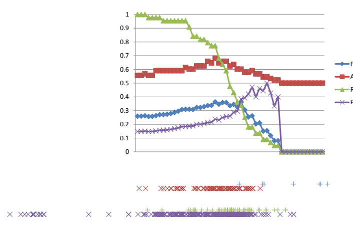
(e) Topic 35



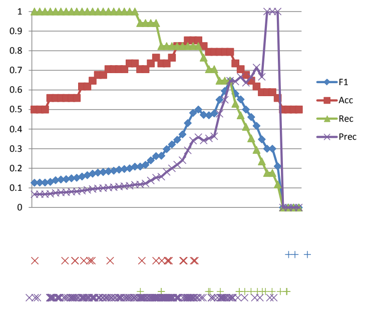
(f) Topic 36



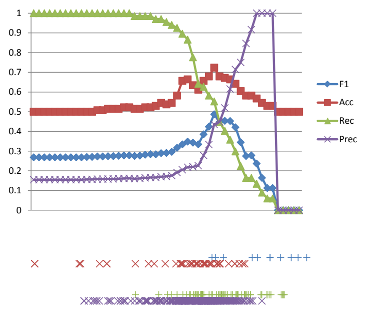
(g) Topic 37



(h) Topic 38

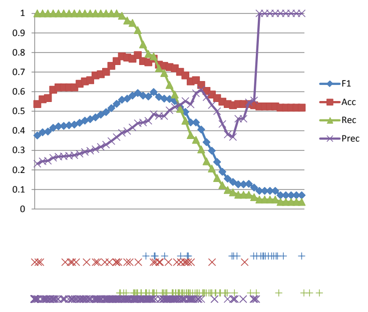


(i) Topic 39

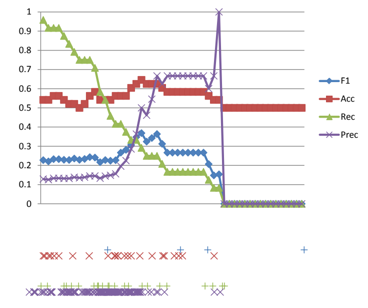


(j) Topic 40

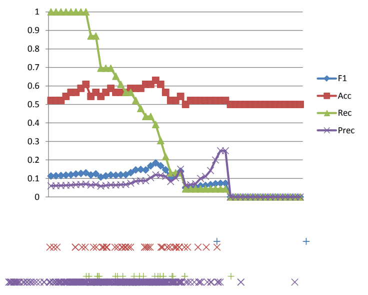
Figure C.5: RFD: performance in training set weights distribution topic 31-40.



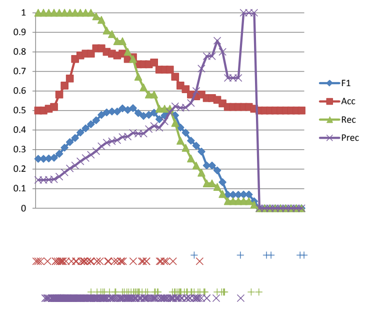
(a) Topic 41



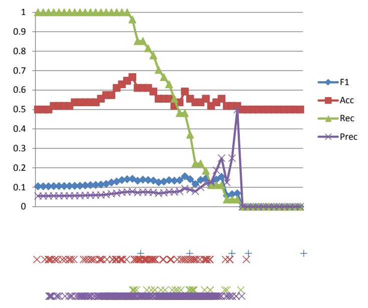
(b) Topic 42



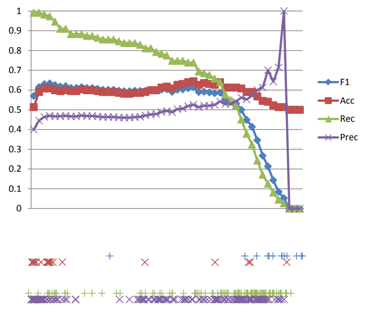
(c) Topic 43



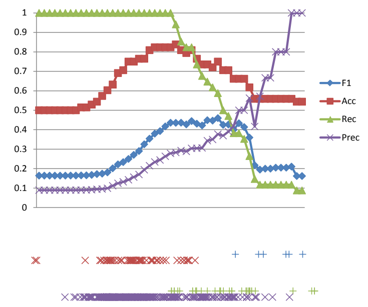
(d) Topic 44



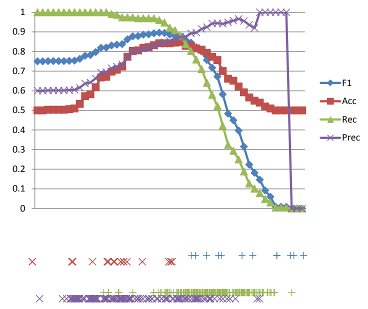
(e) Topic 45



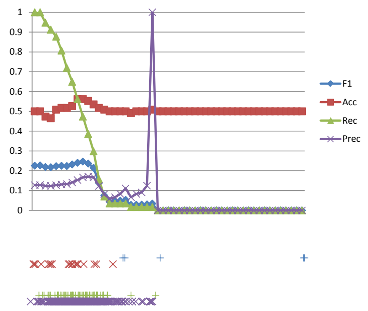
(f) Topic 46



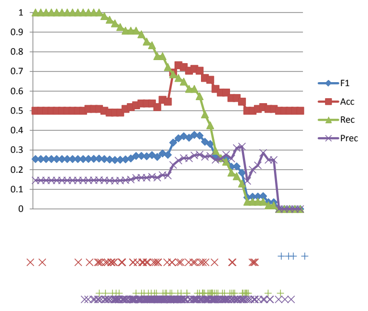
(g) Topic 47



(h) Topic 48



(i) Topic 49



(j) Topic 50

Figure C.6: RFD$_\tau$: performance in training set weights distribution topic 41-50.

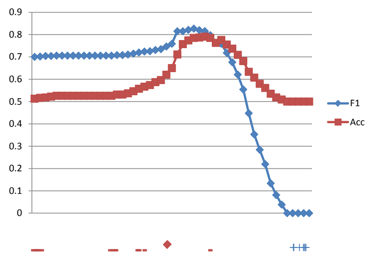
Appendix D

Rocchio Performance in Training Weight Distribution

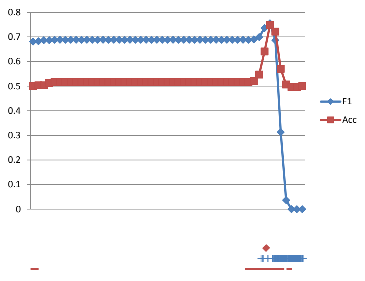
The figures in this appendix show the performance of Rocchio at different points, from the minimal ($min(score(d \in D))$) score to the maximum ($max(score(d \in D))$) score of training documents. The left side of the horizontal axis states the $min(score(d \in D)$ and the right side of the horizontal axis states the $max(score(d \in D)$ of the topic.

We chose Rocchio because its classification uses the decision boundary (default value is zero) and a strong text classifier.

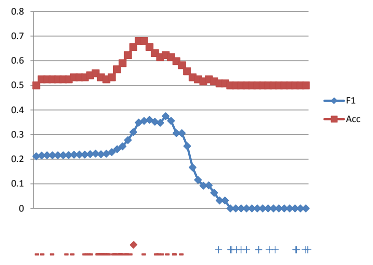
From Figure D.1 to Figure D.5, the symbols under the horizontal axis denote the training document positions based on their score. On the right side, '+' indicates a positive document, and the left side '-' indicates a negative document. The diamond-shaped symbol represents the decision boundary (i.e. the score is zero).



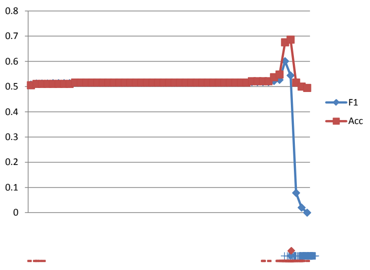
(a) Topic 1



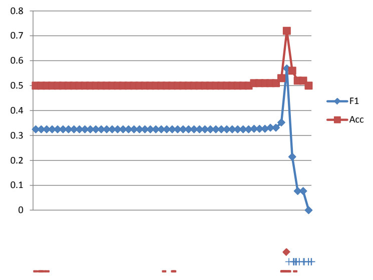
(b) Topic 2



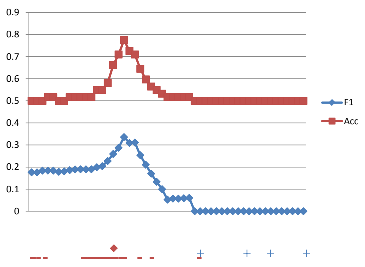
(c) Topic 3



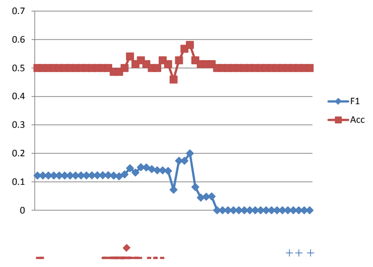
(d) Topic 4



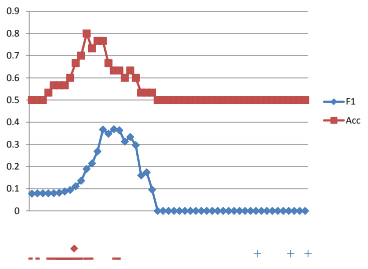
(e) Topic 5



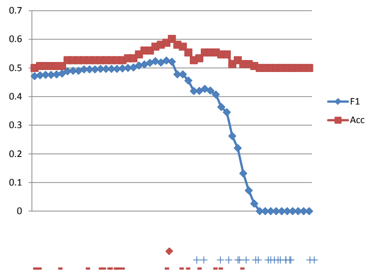
(f) Topic 6



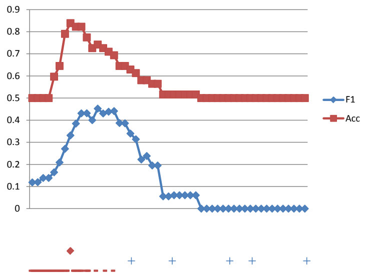
(g) Topic 7



(h) Topic 8

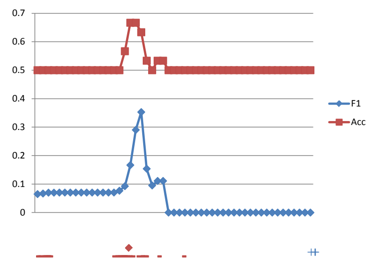


(i) Topic 9

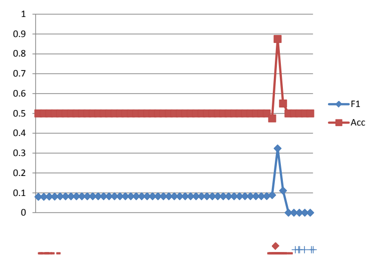


(j) Topic 10

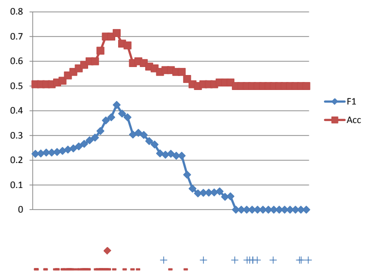
Figure D.1: Rocchio: performance in training set weights distribution topic 1-10.



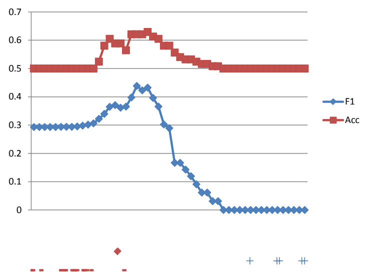
(a) Topic 11



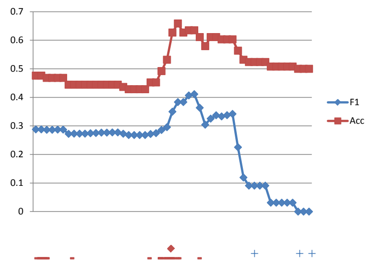
(b) Topic 12



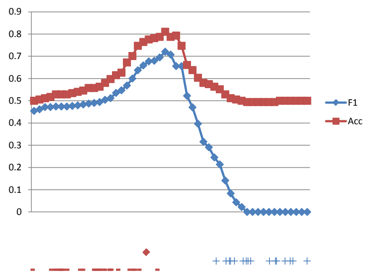
(c) Topic 13



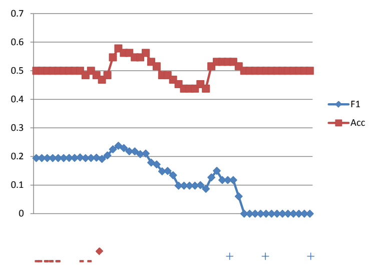
(d) Topic 14



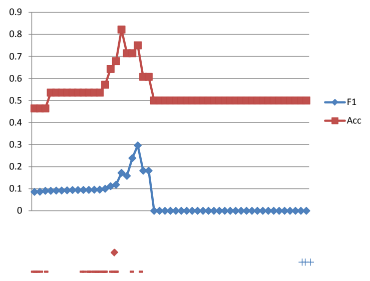
(e) Topic 15



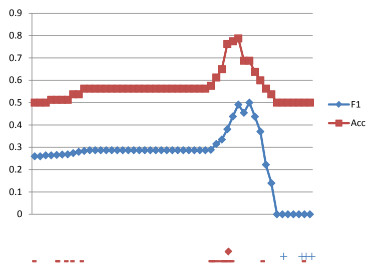
(f) Topic 16



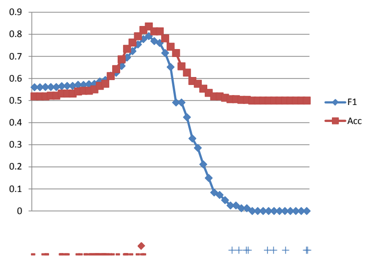
(g) Topic 17



(h) Topic 18

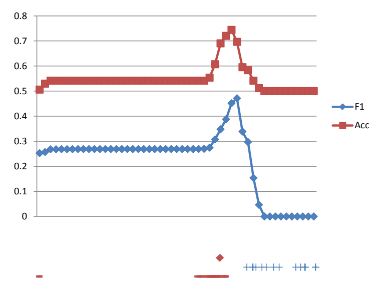


(i) Topic 19

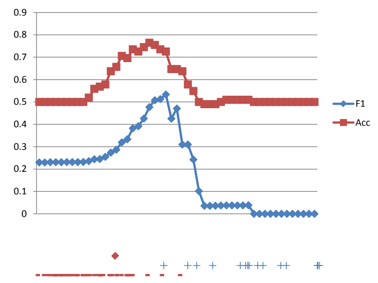


(j) Topic 20

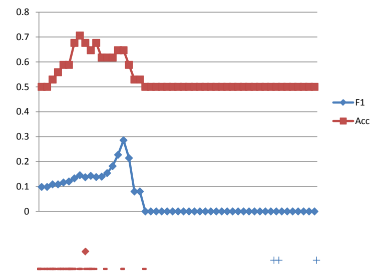
Figure D.2: Rocchio: performance in training set weights distribution topic 11-20.



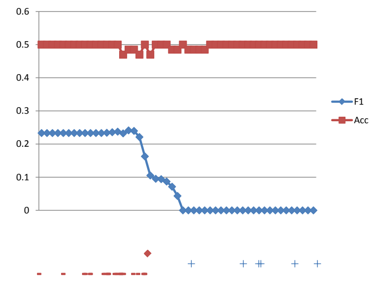
(a) Topic 21



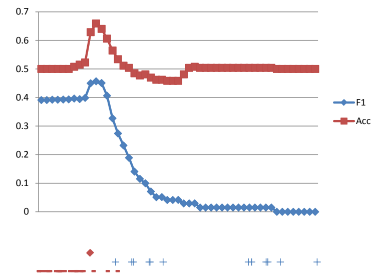
(b) Topic 22



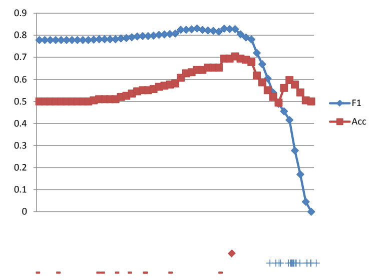
(c) Topic 23



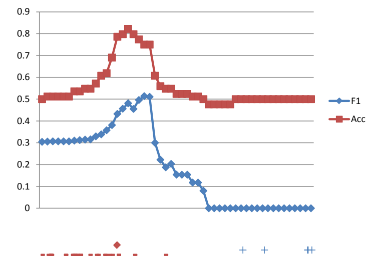
(d) Topic 24



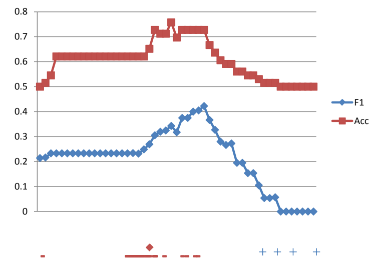
(e) Topic 25



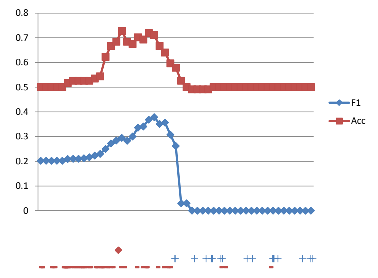
(f) Topic 26



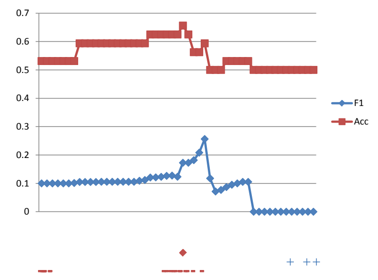
(g) Topic 27



(h) Topic 28

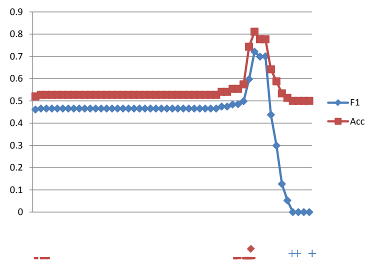


(i) Topic 29

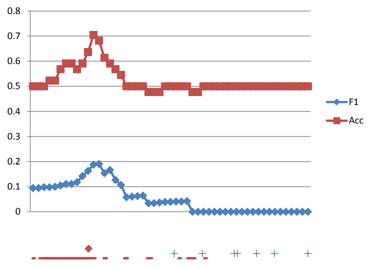


(j) Topic 30

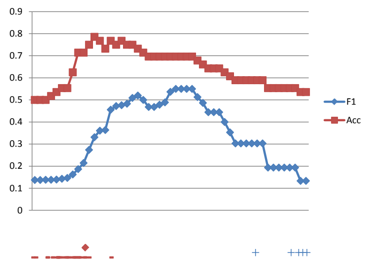
Figure D.3: Rocchio: performance in training set weights distribution topic 21-30.



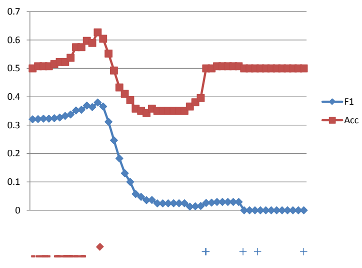
(a) Topic 31



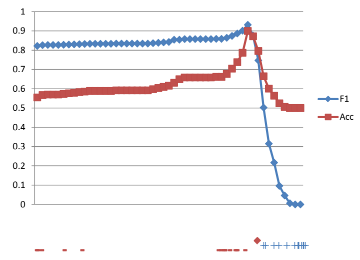
(b) Topic 32



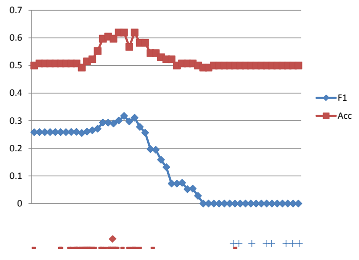
(c) Topic 33



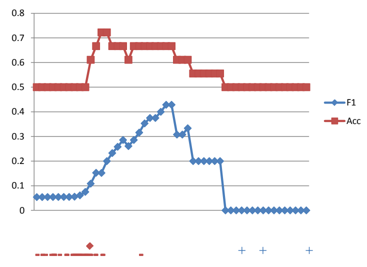
(d) Topic 34



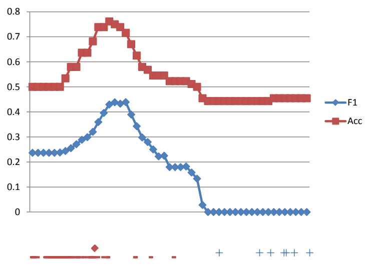
(e) Topic 35



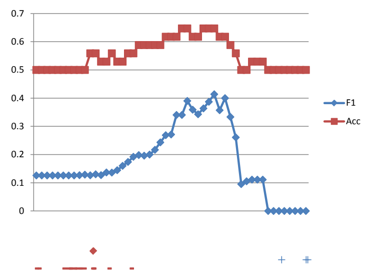
(f) Topic 36



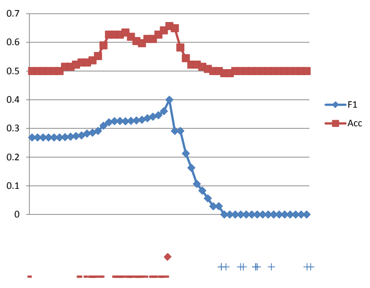
(g) Topic 37



(h) Topic 38

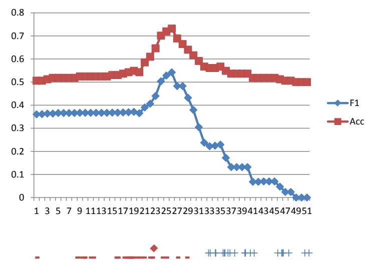


(i) Topic 39

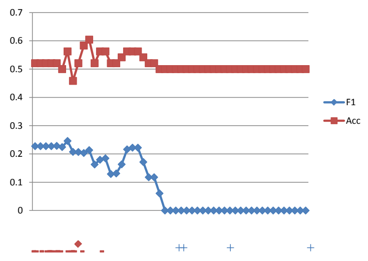


(j) Topic 40

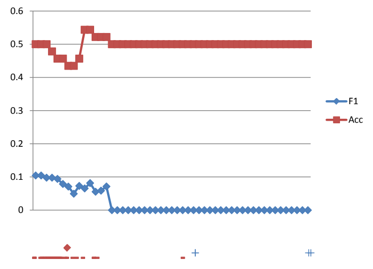
Figure D.4: Rocchio: performance in training set weights distribution topic 31-40.



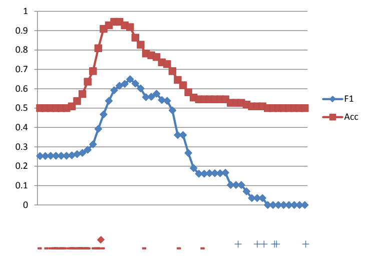
(a) Topic 41



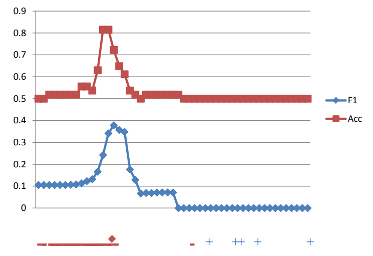
(b) Topic 42



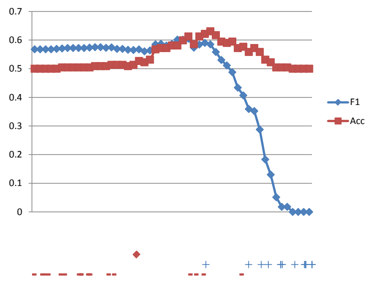
(c) Topic 43



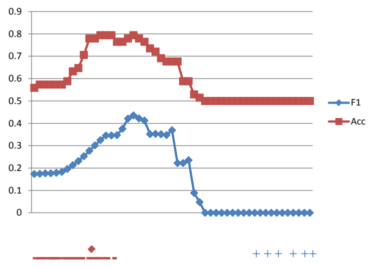
(d) Topic 44



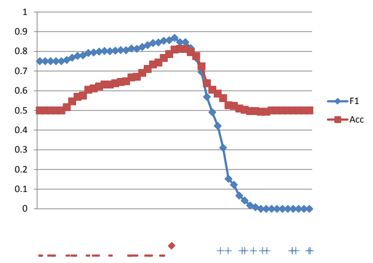
(e) Topic 45



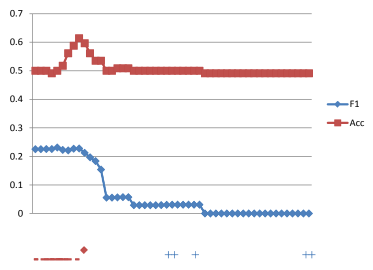
(f) Topic 46



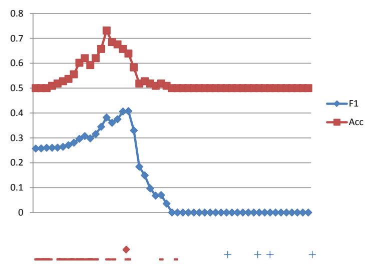
(g) Topic 47



(h) Topic 48



(i) Topic 49



(j) Topic 50

Figure D.5: Rocchio: performance in training set weights distribution topic 41-50.

Appendix E

TP, FN, TN and FN in Classifier Combination Rocchio-RFD$_{\mathcal{T}}$

Table E.1 shows the number of true positive (TP), false pasitive (FP), true negative (TN) and false negative (FN) results for the classifier combination Rocchio-RFD$_{\mathcal{T}}$. The first one is for Rocchio, the second one is for RFD$_{\mathcal{T}}$, and the figures in the brackets show the results for classifier combination, Rocchio-RFD$_{\mathcal{T}}$. For example, a value in column FN-TP(FN) means the number of documents with FN for Rocchio, TP for RFD$_{\mathcal{T}}$ and FN (in brackets) for the final result.

Table E.1: TP, FN, TN and FN in classifier combination Rocchio-RFD$_\tau$.

Topic	TP-TP(TP)	FN-TP(FN)	FP-FP(FP)	TN-FP(TN)	FP-TN(TN)	TN-TN(TN)	TP-FN(FN)	FN-FN(FN)
1	188	0	22	2	165	81	111	8
2	143	9	50	13	31	55	2	5
3	34	1	5	1	210	251	17	9
4	77	4	81	21	49	34	13	0
5	38	0	15	13	95	85	11	1
6	16	3	77	43	73	97	11	1
7	20	0	100	33	284	117	11	6
8	5	0	4	0	224	143	10	0
9	60	3	86	16	23	41	6	5
10	20	0	30	17	55	358	6	5
11	7	0	9	5	163	259	6	2
12	11	0	16	0	326	119	8	1
13	42	7	68	77	124	213	15	6
14	37	17	71	110	73	45	8	0
15	30	14	89	19	81	105	15	4
16	46	0	16	0	66	129	40	1
17	20	1	14	2	213	36	8	3
18	0	0	16	0	151	112	11	3
19	0	0	0	0	127	104	40	0
20	115	9	37	37	28	155	29	5
21	66	10	72	2	209	230	7	1
22	33	8	22	3	179	138	9	1
23	6	0	9	0	143	173	8	3
24	3	2	18	8	80	111	7	21
25	82	17	98	48	145	121	28	5
26	133	2	12	1	41	44	26	11
27	26	0	37	2	64	93	14	2
28	27	0	77	2	96	68	4	2
29	33	1	60	8	190	192	19	4
30	10	1	36	6	136	113	4	1
31	68	0	50	11	41	76	3	3
32	5	3	71	93	104	156	13	1
33	15	0	16	3	133	200	11	2
34	18	9	25	71	128	60	31	9
35	245	58	6	35	0	123	16	18
36	14	1	48	2	178	157	33	19
37	9	0	48	11	105	152	0	0
38	15	1	17	17	138	112	25	3
39	13	1	45	1	171	19	3	0
40	24	15	31	43	45	246	12	16
41	54	6	56	9	131	101	22	0
42	9	1	33	17	61	63	6	8
43	1	0	4	2	149	239	6	16
44	12	0	4	6	125	190	42	1
45	12	10	36	232	13	180	4	1
46	93	0	109	0	38	22	8	10
47	27	1	62	1	101	182	6	0
48	204	4	33	2	21	96	11	9
49	1	1	23	6	65	298	16	39
50	2	0	4	7	33	273	19	33
Total	2169	220	1968	1058	5624	6767	791	304

Bibliography

[1] David W Aha, Dennis Kibler, and Marc K Albert. Instance-based learning algorithms. *Machine learning*, 6(1):37–66, 1991.

[2] Khalid Al-Kofahi, Alex Tyrrell, Arun Vachher, Tim Travers, and Peter Jackson. Combining multiple classifiers for text categorization. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 97–104. ACM, 2001.

[3] Erin L Allwein, Robert E Schapire, and Yoram Singer. Reducing multiclass to binary: A unifying approach for margin classifiers. *The Journal of Machine Learning Research*, 1:113–141, 2001.

[4] John Robert Anderson, Ryszard S Michalski, Ryszard Stanisław Michalski, Jaime Guillermo Carbonell, and Tom Michael Mitchell. *Machine learning: An artificial intelligence approach*, volume 2. Morgan Kaufmann, 1986.

[5] L.D. Baker and A.K. McCallum. Distributional clustering of words for text classification. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 96–103. ACM, 1998.

[6] R. Bekkerman and M. Gavish. High-precision phrase-based document classification on a modern scale. In *Proceedings of the 17th ACM SIGKDD*

international conference on Knowledge discovery and data mining, pages 231–239. ACM, 2011.

[7] R. Bekkerman, R. El-Yaniv, N. Tishby, and Y. Winter. Distributional word clusters vs. words for text categorization. *The Journal of Machine Learning Research*, 3:1183–1208, 2003.

[8] Nicholas J Belkin and W Bruce Croft. Information filtering and information retrieval: two sides of the same coin? *Communications of the ACM*, 35(12): 29–38, 1992.

[9] D.A. Bell, JW Guan, and Y. Bi. On combining classifier mass functions for text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 17(10):1307–1319, 2005.

[10] Kristin P Bennett and Colin Campbell. Support vector machines: hype or hallelujah? *ACM SIGKDD Explorations Newsletter*, 2(2):1–13, 2000.

[11] Paul N Bennett, Susan T Dumais, and Eric Horvitz. The combination of text classifiers using reliability indicators. *Information Retrieval*, 8(1):67–100, 2005.

[12] P.N. Bennett. Assessing the calibration of naive bayes posterior estimates. Technical report, Computer Science Department, Carniege Melon University, 2000.

[13] Yaxin Bi, David Bell, Hui Wang, Gongde Guo, and Kieran Greer. Combining multiple classifiers using dempsterŠs rule of combination for text categorization. In *Modeling Decisions for Artificial Intelligence*, pages 127–138. Springer, 2004.

[14] Moch Arif Bijaksana, Yuefeng Li, and Abdulmohsen Algarni. A pattern based two-stage text classifier. In *Machine Learning and Data Mining in Pattern Recognition*, pages 169–182. Springer, 2013.

[15] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[16] Joan Fisher Box. Guinness, gosset, fisher, and small samples. *Statistical Science*, 2(1):45–52, 1987.

[17] Leo Breiman. Bagging predictors. *Machine learning*, 24(2):123–140, 1996.

[18] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[19] Leo Breiman, Jerome Friedman, Charles J Stone, and Richard A Olshen. *Classification and regression trees*. CRC press, 1984.

[20] Leonard A Breslow and David W Aha. Simplifying decision trees: A survey. *The Knowledge Engineering Review*, 12(01):1–40, 1997.

[21] C. Buckley and E.M. Voorhees. Evaluating evaluation measure stability. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 33–40. ACM, 2000.

[22] Wray Buntine. Learning classification trees. *Statistics and computing*, 2 (2):63–73, 1992.

[23] Christopher JC Burges. A tutorial on support vector machines for pattern recognition. *Data mining and knowledge discovery*, 2(2):121–167, 1998.

[24] Toon Calders and Sicco Verwer. Three naive bayes approaches for discrimination-free classification. *Data Mining and Knowledge Discovery*, 21(2):277–292, 2010.

[25] M.F. Caropreso, S. Matwin, and F. Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. *Text databases and document management: Theory and practice*, pages 78–102, 2001.

[26] Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *Proceedings of the 23rd international conference on Machine learning*, pages 161–168. ACM, 2006.

[27] Weizhu Chen, Jun Yan, Benyu Zhang, Zheng Chen, and Qiang Yang. Document transformation for multi-label feature selection in text categorization. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on*, pages 451–456. IEEE, 2007.

[28] Peter Clark and Robin Boswell. Rule induction with cn2: Some recent improvements. In *Machine learningŮEWSL-91*, pages 151–163. Springer, 1991.

[29] William W Cohen. Learning to classify english text with ilp methods. *Advances in inductive logic programming*, 32:124–143, 1995.

[30] William W Cohen and Haym Hirsh. Joins that generalize: Text classification using whirl. In *KDD*, pages 169–173, 1998.

[31] William W Cohen and Yoram Singer. Context-sensitive learning methods for text categorization. *ACM Transactions on Information Systems (TOIS)*, 17(2):141–173, 1999.

[32] Gordon V Cormack. Email spam filtering: A systematic review. *Foundations and Trends in Information Retrieval*, 1(4):335–455, 2007.

[33] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[34] Robert H Creecy, Brij M Masand, Stephen J Smith, and David L Waltz. Trading mips and memory for knowledge engineering. *Communications of the ACM*, 35(8):48–64, 1992.

[35] W.B. Croft, D. Metzler, and T. Strohman. *Search engines: Information retrieval in practice*. Addison-Wesley, 2010.

[36] Ido Dagan, Yael Karov, Dan Roth, et al. Mistake-driven learning in text categorization. In *Proceedings of the second conference on empirical methods in NLP*, pages 55–63, 1997.

[37] F. Debole and F. Sebastiani. An analysis of the relative hardness of reuters-21578 subsets. *Journal of the American Society for Information Science and Technology*, 56(6):584–596, 2005.

[38] S. Deerwester, S.T. Dumais, G.W. Furnas, T.K. Landauer, and R. Harshman. Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407, 1990.

[39] Luca Didaci, Giorgio Fumera, and Fabio Roli. Diversity in classifier ensembles: fertile concept or dead end? In *Multiple Classifier Systems*, pages 37–48. Springer, 2013.

[40] Thomas G Dietterich. Ensemble methods in machine learning. In *Multiple classifier systems*, pages 1–15. Springer, 2000.

[41] Thomas G Dietterich and Ghulum Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2(263):286, 1995.

[42] Pedro Domingos and Michael Pazzani. On the optimality of the simple bayesian classifier under zero-one loss. *Machine learning*, 29(2-3):103–130, 1997.

[43] S.T. Dumais, G.W. Furnas, T.K. Landauer, S. Deerwester, and R. Harshman. Using latent semantic analysis to improve access to textual information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–285. ACM, 1988.

[44] Susan Dumais and Hao Chen. Hierarchical classification of web content. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 256–263. ACM, 2000.

[45] Susan Dumais, John Platt, David Heckerman, and Mehran Sahami. Inductive learning algorithms and representations for text categorization. In *Proceedings of the seventh international conference on Information and knowledge management*, pages 148–155. ACM, 1998.

[46] A. Elisseeff and J. Weston. Kernel methods for multi-labelled classification and categorical regression problems. *Advances in neural information processing systems*, 14:681–687, 2002.

[47] Floriana Esposito, Donato Malerba, Giovanni Semeraro, and J Kay. A comparative analysis of methods for pruning decision trees. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(5):476–491, 1997.

[48] Eibe Frank and Ian H Witten. Generating accurate rule sets without global optimization. In *Proceedings of the Fifteenth International Conference on Machine Learning*, pages 144–151. Morgan Kaufmann Publishers Inc., 1998.

[49] D Freedman, R Pisani, and R Purves. *Statistics 4th.* New York, NY: WW Norton & Company, Ltd, 2007.

[50] Yoav Freund and Robert E Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. *Journal of computer and system sciences*, 55(1):119–139, 1997.

[51] G.P.C. Fung, J.X. Yu, H. Lu, and P.S. Yu. Text classification without negative examples revisit. *IEEE Transactions on Knowledge and Data Engineering*, 18(1):6–20, 2006.

[52] J. Furnkranz, T. Mitchell, E. Riloff, et al. A case study in using linguistic phrases for text categorization on the www. In *Proceedings from the AAAI/ICML Workshop on Learning for Text Categorization*, pages 5–12, 1998.

[53] Johannes Furnkranz and Gerhard Widmer. Incremental reduced error pruning. In *International Conference on Machine Learning*, pages 70–77, 1994.

[54] Alexander Genkin, David D Lewis, and David Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49(3): 291–304, 2007.

[55] R. Ghani, S. Slattery, and Y. Yang. Hypertext categorization using hyperlink patterns and meta data. In *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning*, pages 178–185, 2001.

[56] Rayid Ghani. Using error-correcting codes for text classification. In *ICML*, pages 303–310, 2000.

[57] Shantanu Godbole and Sunita Sarawagi. Discriminative methods for multi-labeled classification. In *Advances in Knowledge Discovery and Data Mining*, pages 22–30. Springer, 2004.

[58] S. Gopal and Y. Yang. Multilabel classification with meta-level features. In *Proceeding of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 315–322. ACM, 2010.

[59] Frank E Grubbs. Procedures for detecting outlying observations in samples. *Technometrics*, 11(1):1–21, 1969.

[60] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. The weka data mining software: an update. *ACM SIGKDD Explorations Newsletter*, 11(1):10–18, 2009.

[61] E.H. Han and G. Karypis. Centroid-based document classification: Analysis and experimental results. *Principles of Data Mining and Knowledge Discovery*, pages 116–123, 2000.

[62] Jiawei Han, Micheline Kamber, and Jian Pei. *Data mining: concepts and techniques*. Morgan kaufmann, 2006.

[63] Jiawei Han, Hong Cheng, Dong Xin, and Xifeng Yan. Frequent pattern mining: current status and future directions. *Data Mining and Knowledge Discovery*, 15(1):55–86, 2007.

[64] Lars Kai Hansen and Peter Salamon. Neural network ensembles. *IEEE transactions on pattern analysis and machine intelligence*, 12:993–1001, 1990.

[65] Trevor Hastie and Robert Tibshirani. Classification by pairwise coupling. *The annals of statistics*, 26(2):451–471, 1998.

[66] Marti A. Hearst, ST Dumais, E Osman, John Platt, and Bernhard Scholkopf. Support vector machines. *Intelligent Systems and their Applications, IEEE*, 13(4):18–28, 1998.

[67] David Heckerman. Bayesian networks for data mining. *Data mining and knowledge discovery*, 1(1):79–119, 1997.

[68] David Heckerman, Dan Geiger, and David M Chickering. Learning bayesian networks: The combination of knowledge and statistical data. *Machine learning*, 20(3):197–243, 1995.

[69] Robert C Holte. Very simple classification rules perform well on most commonly used datasets. *Machine learning*, 11(1):63–90, 1993.

[70] David Hull. Improving text retrieval for the routing problem using latent semantic indexing. In *SIGIRŠ94*, pages 282–291. Springer, 1994.

[71] G. Ifrim, G. Bakir, and G. Weikum. Fast logistic regression for text categorization with variable-length n-grams. In *Proceeding of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 354–362. ACM, 2008.

[72] David J Ittner, David D Lewis, and David D Ahn. Text categorization of low quality images. In *Symposium on Document Analysis and Information Retrieval*, pages 301–315. Citeseer, 1995.

[73] Makoto Iwayama and Takenobu Tokunaga. Cluster-based text categorization: a comparison of category search strategies. In *Proceedings of the 18th*

annual international ACM SIGIR conference on Research and development in information retrieval, pages 273–280. ACM, 1995.

[74] Nathalie Japkowicz and Shaju Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[75] Richard Jensen and Qiang Shen. Fuzzy–rough attribute reduction with application to web categorization. *Fuzzy sets and systems*, 141(3):469–485, 2004.

[76] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. *Machine Learning: ECML-98*, pages 137–142, 1998.

[77] Thorsten Joachims. Transductive inference for text classification using support vector machines. In *ICML*, volume 99, pages 200–209, 1999.

[78] George H John and Pat Langley. Estimating continuous distributions in bayesian classifiers. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 338–345. Morgan Kaufmann Publishers Inc., 1995.

[79] Anna Jurek, Yaxin Bi, Shengli Wu, and Chris Nugent. A survey of commonly used ensemble-based classification techniques. *The Knowledge Engineering Review*, pages 1–31, 2013.

[80] David R Karger and Matthias Ruhl. Finding nearest neighbors in growth-restricted metrics. In *Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*, pages 741–750. ACM, 2002.

[81] Gordon V Kass. An exploratory technique for investigating large quantities of categorical data. *Applied statistics*, pages 119–127, 1980.

[82] A. Kehagias, V. Petridis, V.G. Kaburlasos, and P. Fragkou. A comparison of word-and sense-based text categorization using several classification algorithms. *Journal of Intelligent Information Systems*, 21(3):227–247, 2003.

[83] Daijin Kim. Data classification based on tolerant rough set. *Pattern recognition*, 34(8):1613–1624, 2001.

[84] Ralf Klinkenberg and Thorsten Joachims. Detecting concept drift with support vector machines. In *ICML*, pages 487–494, 2000.

[85] Anders Krogh, Jesper Vedelsby, et al. Neural network ensembles, cross validation, and active learning. *Advances in neural information processing systems*, pages 231–238, 1995.

[86] Ludmila I Kuncheva and Christopher J Whitaker. Measures of diversity in classifier ensembles and their relationship with the ensemble accuracy. *Machine learning*, 51(2):181–207, 2003.

[87] Savio LY Lam and Dik Lun Lee. Feature reduction for neural network based text categorization. In *Database Systems for Advanced Applications, 1999. Proceedings., 6th International Conference on*, pages 195–202. IEEE, 1999.

[88] M. Lan, C.L. Tan, J. Su, and Y. Lu. Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):721–735, 2009.

[89] Man Lan, Chew-Lim Tan, Hwee-Boon Low, and Sam-Yuan Sung. A comprehensive comparative study on term weighting schemes for text categorization with support vector machines. In *Special interest tracks and posters of the 14th international conference on World Wide Web*, pages 1032–1033. ACM, 2005.

[90] GH Landeweerd, T Timmers, Edzard S Gelsema, M Bins, and MR Halie. Binary tree versus single level tree classification of white blood cells. *Pattern Recognition*, 16(6):571–577, 1983.

[91] Pat Langley et al. An analysis of bayesian classifiers. In *Proceedings of the tenth national conference on Artificial intelligence*, pages 223–228. AAAI Press, 1992.

[92] Leah S Larkey and W Bruce Croft. Combining classifiers in text categorization. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 289–297. ACM, 1996.

[93] David D Lewis. Naive (bayes) at forty: The independence assumption in information retrieval. In *Machine learning: ECML-98*, pages 4–15. Springer, 1998.

[94] D.D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–50. ACM, 1992.

[95] D.D. Lewis. *Representation and learning in information retrieval.* PhD thesis, University of Massachusetts, 1992.

[96] D.D. Lewis, Y. Yang, T.G. Rose, and F. Li. Rcv1: A new benchmark collection for text categorization research. *The Journal of Machine Learning Research*, 5:361–397, 2004.

[97] Hang Li and Kenji Yamanishi. Text classification using esc-based stochastic decision lists. In *Proceedings of the eighth international conference on Information and knowledge management*, pages 122–130. ACM, 1999.

[98] Y. Li and N. Zhong. Mining ontology for automatically acquiring web user information needs. *IEEE Transactions on Knowledge and Data Engineering*, 18(4):554–568, 2006.

[99] Y. Li, A. Algarni, and N. Zhong. Mining positive and negative patterns for relevance feature discovery. In *Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 753–762. ACM, 2010.

[100] Yan Li, Simon Chi-Keung Shiu, Sankar K Pal, and James Nga-Kwok Liu. A rough set-based case-based reasoner for text categorization. *International journal of approximate reasoning*, 41(2):229–255, 2006.

[101] Yong H Li and Anil K. Jain. Classification of text documents. *The Computer Journal*, 41(8):537–546, 1998.

[102] Yuefeng Li, Xujuan Zhou, Peter Bruza, Yue Xu, and Raymond Y.K. Lau. A two-stage text mining model for information filtering. In *Proceedings of the 17th ACM conference on Information and knowledge management*, pages 1023–1032. ACM, 2008.

[103] Yuefeng Li, Xujuan Zhou, Peter Bruza, Yue Xu, and Raymond YK Lau. A two-stage decision model for information filtering. *Decision Support Systems*, 52(3):706–716, 2011.

[104] Chia-Chi Liao and Kuo-Wei Hsu. A rule-based classification algorithm: A rough set approach. In *Computational Intelligence and Cybernetics (CyberneticsCom), 2012 IEEE International Conference on*, pages 1–5. IEEE, 2012.

[105] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press Cambridge, 2008.

[106] Vangelis Metsis, Ion Androutsopoulos, and Georgios Paliouras. Spam filtering with naive bayes-which naive bayes? In *CEAS*, pages 27–28, 2006.

[107] Duoqian Miao, Qiguo Duan, Hongyun Zhang, and Na Jiao. Rough set based hybrid algorithm for text classification. *Expert Systems with Applications*, 36(5):9168–9174, 2009.

[108] G.A. Miller et al. Wordnet: a lexical database for english. *Communications of the ACM*, 38(11):39–41, 1995.

[109] Bernard ME Moret. Decision trees and diagrams. *ACM Computing Surveys (CSUR)*, 14(4):593–623, 1982.

[110] A. Moschitti. Syntactic and semantic kernels for short text pair categorization. In *Proceedings of the 12th Conference of the European Chapter of the ACL (EACL 2009)*, pages 576–584, 2009.

[111] A. Moschitti and R. Basili. Complex linguistic features for text classification: A comprehensive study. *Advances in Information Retrieval*, pages 181–196, 2004.

[112] Isabelle Moulinier and Jean-Gabriel Ganascia. Applying an existing machine learning algorithm to text categorization. In *Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing*, pages 343–354. Springer, 1996.

[113] Isabelle Moulinier, Gailius Raskinis, and J Ganascia. Text categorization: a symbolic approach. In *proceedings of the fifth annual symposium on document analysis and information retrieval*, pages 87–99, 1996.

[114] Sreerama K Murthy. Automatic construction of decision trees from data:

A multi-disciplinary survey. *Data mining and knowledge discovery*, 2(4): 345–389, 1998.

[115] Balas Kausik Natarajan. Machine learning:{A} theoretical approach. 1991.

[116] A. Özgür, L. Özgür, and T. Güngör. Text categorization with class-based and corpus-based keyword selection. *Computer and Information Sciences-ISCIS 2005*, pages 606–615, 2005.

[117] L. Özgür and T. Güngör. Text classification with the support of pruned dependency patterns. *Pattern Recognition Letters*, 31(12):1598–1607, 2010.

[118] Sankar K Pal and Sushmita Mitra. Multilayer perceptron, fuzzy sets, and classification. *IEEE Transactions on Neural Networks*, 3(5):683–697, 1992.

[119] Zhongdang Pan and Gerald M Kosicki. Framing analysis: An approach to news discourse. *Political communication*, 10(1):55–75, 1993.

[120] Krishna R Pattipati and Mark G Alexandridis. Application of heuristic search and information theory to sequential fault diagnosis. *Systems, Man and Cybernetics, IEEE Transactions on*, 20(4):872–887, 1990.

[121] Zdzisław Pawlak. Rough sets. *International Journal of Computer & Information Sciences*, 11(5):341–356, 1982.

[122] Zdzisław Pawlak. A rough set view on bayes' theorem. *International Journal of Intelligent Systems*, 18(5):487–498, 2003.

[123] X. Peng and B. Choi. Document classifications based on word semantic hierarchies. In *Proceedings of the International Conference on Artificial Intelligence and Applications (AIA05)*, pages 362–367, 2005.

[124] F. Pereira, N. Tishby, and L. Lee. Distributional clustering of english words. In *Proceedings of the 31st annual meeting on Association for Computational Linguistics*, pages 183–190. Association for Computational Linguistics, 1993.

[125] John C Platt. Fast training of support vector machines using sequential minimal optimization. In *Advances in kernel methods*, pages 185–208. MIT Press, 1999.

[126] J. Ross Quinlan. Simplifying decision trees. *International journal of man-machine studies*, 27(3):221–234, 1987.

[127] J Ross Quinlan et al. *Discovering rules by induction from large collections of examples*. Expert systems in the micro electronic age. Edinburgh University Press, 1979.

[128] John Ross Quinlan. *C4.5: programs for machine learning*, volume 1. Morgan kaufmann, 1993.

[129] Marco Ramoni and Paola Sebastiani. Robust bayes classifiers. *Artificial Intelligence*, 125(1):209–226, 2001.

[130] Irina Rish. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, pages 41–46, 2001.

[131] J.J. Rocchio. Relevance feedback in information retrieval. *SMART Retrieval System Experimens in Automatic Document Processing*, pages 313–323, 1971.

[132] Lior Rokach and Oded Maimon. Top-down induction of decision trees

classifiers-a survey. *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, 35(4):476–487, 2005.

[133] Miguel E Ruiz and Padmini Srinivasan. Hierarchical neural networks for text categorization (poster abstract). In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 281–282. ACM, 1999.

[134] S Rasoul Safavian and David Landgrebe. A survey of decision tree classifier methodology. *IEEE transactions on systems, man, and cybernetics*, 21 (3):660–674, 1991.

[135] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5): 513–523, 1988.

[136] Manish Sarkar. Fuzzy-rough nearest neighbor algorithms in classification. *Fuzzy Sets and Systems*, 158(19):2134–2152, 2007.

[137] Cullen Schaffer. Overfitting avoidance as bias. *Machine learning*, 10(2): 153–178, 1993.

[138] R.E. Schapire, Y. Singer, and A. Singhal. Boosting and rocchio applied to text filtering. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 215–223. ACM, 1998.

[139] Robert E Schapire. The boosting approach to machine learning: An overview. In *Nonlinear estimation and classification*, pages 149–171. Springer, 2003.

[140] Robert E Schapire and Yoram Singer. Boostexter: A boosting-based system for text categorization. *Machine learning*, 39(2):135–168, 2000.

[141] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. MIT press, 2002.

[142] H. Schütze, D.A. Hull, and J.O. Pedersen. A comparison of classifiers and document representations for the routing problem. In *Proceedings of the 18th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 229–237. ACM, 1995.

[143] S. Scott and S. Matwin. Feature engineering for text classification. In *Proceedings of International Conference on Machine Learning 1999*, pages 379–388. Citeseer, 1999.

[144] F. Sebastiani. Machine learning in automated text categorization. *ACM computing surveys (CSUR)*, 34(1):1–47, 2002.

[145] Terrence J Sejnowski and Charles R Rosenberg. Parallel networks that learn to pronounce english text. *Complex systems*, 1(1):145–168, 1987.

[146] Nist Sematech. *Engineering statistics handbook*. NIST SEMATECH, 2006.

[147] James G Shanahan and Norbert Roma. Boosting support vector machines for text classification through parameter-free threshold relaxation. In *Proceedings of the twelfth international conference on Information and knowledge management*, pages 247–254. ACM, 2003.

[148] James G Shanahan and Norbert Roma. Improving svm text classification

performance through threshold adjustment. In *Machine Learning: ECML 2003*, pages 361–372. Springer, 2003.

[149] S. Shehata, F. Karray, and M. Kamel. A concept-based model for enhancing text categorization. In *Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 629–637. ACM, 2007.

[150] D. Shen, J.T. Sun, Q. Yang, H. Zhao, and Z. Chen. Text classification improved through automatically extracted sequences. In *Data Engineering, 2006. ICDE'06. Proceedings of the 22nd International Conference on*, pages 121–121. IEEE, 2006.

[151] N. Slonim and N. Tishby. The power of word clusters for text classification. In *23rd European Colloquium on Information Retrieval Research*, volume 1, 2001.

[152] I. Soboroff and S. Robertson. Building a filtering test collection for trec 2002. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 243–250. ACM, 2003.

[153] Pascal Soucy and Guy W Mineau. A simple knn algorithm for text categorization. In *Data Mining, 2001. ICDM 2001, Proceedings IEEE International Conference on*, pages 647–648. IEEE, 2001.

[154] A. Stavrianou, P. Andritsos, and N. Nicoloyannis. Overview and semantic issues of text mining. *ACM Sigmod Record*, 36(3):23–34, 2007.

[155] Pang-Ning Tan et al. *Introduction to data mining*. Pearson Education, 2007.

[156] S. Tan. An effective refinement strategy for knn text classifier. *Expert Systems with Applications*, 30(2):290–298, 2006.

[157] Songbo Tan and Xueqi Cheng. Using hypothesis margin to boost centroid text classifier. In *Proceedings of the 2007 ACM symposium on Applied computing*, pages 398–403. ACM, 2007.

[158] Songbo Tan, Xueqi Cheng, Moustafa M Ghanem, Bin Wang, and Hongbo Xu. A novel refinement approach for text categorization. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 469–476. ACM, 2005.

[159] David MJ Tax and Robert PW Duin. Using two-class classifiers for multi-class classification. In *Pattern Recognition, 2002. Proceedings. 16th International Conference on*, volume 2, pages 124–127. IEEE, 2002.

[160] Robert Tibshirani. Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, pages 267–288, 1996.

[161] N. Tishby, F.C. Pereira, and W. Bialek. The information bottleneck method. *arXiv preprint physics/0004057*, 2000.

[162] Ioannis Tsochantaridis, Thorsten Joachims, Thomas Hofmann, Yasemin Altun, and Yoram Singer. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6 (9), 2005.

[163] Grigorios Tsoumakas and Ioannis Katakis. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining (IJDWM)*, 3(3):1–13, 2007.

[164] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas. Mining multi-label data. *Data mining and knowledge discovery handbook*, pages 667–685, 2010.

[165] Kagan Tumer and Joydeep Ghosh. Error correlation and error reduction in ensemble classifiers. *Connection science*, 8(3-4):385–404, 1996.

[166] Leslie G Valiant. A theory of the learnable. *Communications of the ACM*, 27(11):1134–1142, 1984.

[167] V. Vapnik. *Statistical learning theory. 1998*. Wiley, New York, 1998.

[168] Vladimir Vapnik. *The nature of statistical learning theory*. springer, 2000.

[169] K. Woodford et al. *Cambridge Learner's Dictionary*. Cambridge University Press, 2008.

[170] S.T. Wu, Y. Li, Y. Xu, B. Pham, and P. Chen. Automatic pattern-taxonomy extraction for web mining. In *Proceedings of IEEE/WIC/ACM International Conference on Web Intelligence, 2004. WI 2004.*, pages 242–248. IEEE, 2004.

[171] S.T. Wu, Y. Li, and Y. Xu. Deploying approaches for pattern refinement in text mining. In *Proceedings of ICDM'06. Sixth International Conference on Data Mining.*, pages 1157–1161, 2006.

[172] T.F. Wu, C.J. Lin, and R.C. Weng. Probability estimates for multi-class classification by pairwise coupling. *The Journal of Machine Learning Research*, 5:975–1005, 2004.

[173] X.B. Xue and Z.H. Zhou. Distributional features for text categorization. *IEEE Transactions on Knowledge and Data Engineering*, 21(3):428–442, 2009.

[174] Y. Yang. An evaluation of statistical approaches to text categorization. *Information retrieval*, 1(1):69–90, 1999.

[175] Y. Yang. A study of thresholding strategies for text categorization. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 137–145. ACM, 2001.

[176] Y. Yang, S. Slattery, and R. Ghani. A study of approaches to hypertext categorization. *Journal of Intelligent Information Systems*, 18(2):219–241, 2002.

[177] Yiming Yang and Xin Liu. A re-examination of text categorization methods. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 42–49. ACM, 1999.

[178] Yiming Yang and Jan O Pedersen. A comparative study on feature selection in text categorization. In *ICML*, volume 97, pages 412–420, 1997.

[179] Yiming Yang, Tom Ault, and Thomas Pierce. Combining multiple learning strategies for effective cross validation. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 1167–1174. Morgan Kaufmann Publishers Inc., 2000.

[180] YY Yao and S. K. Michael Wong. A decision theoretic framework for approximating concepts. *International Journal of Man-machine Studies*, 37(6):793–809, 1992.

[181] B. Zadrozny and C. Elkan. Transforming classifier scores into accurate multiclass probability estimates. In *Proceedings of the eighth ACM*

SIGKDD international conference on Knowledge discovery and data mining, pages 694–699. ACM, 2002.

[182] Sarah Zelikovitz and Haym Hirsh. Using lsi for text classification in the presence of background text. In *Proceedings of the tenth international conference on Information and knowledge management*, pages 113–118. ACM, 2001.

[183] Chengxiang Zhai, Peter Jansen, Emilia Stoica, Norbert Grot, and David A Evans. Threshold calibration in clarit adaptive filtering. In *TREC*, pages 96–103, 1998.

[184] Libiao Zhang, Yuefeng Li, Chao Sun, and Wanvimol Nadee. Rough set based approach to text classification. In *Web Intelligence (WI) and Intelligent Agent Technologies (IAT), 2013 IEEE/WIC/ACM International Joint Conferences on*, volume 3, pages 245–252. IEEE, 2013.

[185] Min-Ling Zhang and Zhi-Hua Zhou. Ml-knn: A lazy learning approach to multi-label learning. *Pattern recognition*, 40(7):2038–2048, 2007.

[186] Tong Zhang and Frank J Oles. Text categorization based on regularized linear classification methods. *Information retrieval*, 4(1):5–31, 2001.

[187] Wenqing Zhao and Zili Zhang. An email classification model based on rough set theory. In *Proceedings of the 2005 International Conference on Active Media Technology, 2005.(AMT 2005)*, pages 403–408. IEEE, 2005.

[188] N. Zhong, Y. Li, and S.T. Wu. Effective pattern discovery for text mining. *Knowledge and Data Engineering, IEEE Transactions on*, 24(1):30–44, 2012.

[189] X. Zhou, Y. Li, P. Bruza, Y. Xu, and R. Lau. Pattern mining for a two-stage information filtering system. *Advances in Knowledge Discovery and Data Mining, PAKDD 2011*, pages 363–374, 2011.

[190] Wojciech Ziarko. Rough set approaches for discovery of rules and attribute dependencies. *Handbook of data mining and knowledge discovery*, pages 328–338, 2002.

[191] Wojciech P Ziarko. *Rough sets, fuzzy sets and knowledge discovery*, volume 15. Springer-Verlag London, 1994.