NTNU

Norwegian University of
Science and Technology

Feature Selection for Text Categorisation

Øystein Løhre Garnes

Master of Science in Computer Science
Submission date: June 2009
Supervisor: Kjetil Nørvåg, IDI
Co-supervisor: Robert Neumayer, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Problem Description

Information retrieval and text mining methods operate on the terms found in text documents. As such, every term found in a collection is analysed and used for further processing. The process of feature selection is performed in order to reduce the number of terms to be used in further analysis (i.e. to identify the most important terms beforehand). The task of this project is to compare a range of feature selection techniques with the goal of a thorough performance evaluation on different test collections.

Assignment given: 15. January 2009
Supervisor: Kjetil Nørvåg, IDI

Preface

This report is the result of the last semester of a five-year study of Computer Science at the Department of Computer and Information Science, Faculty of Information Technology, Mathematics and Electrical Engineering at the Norwegian University of Science and Technology (NTNU) in Trondheim, Norway.

Since my project work from the previous semester was on an entirely different topic, all work is done in one semester. While Text Categorization and indeed Feature Selection was unploughed ground before starting this work, I am glad I took the challenge. It has given me valuable insight in natural language processing, information theory, statistics and machine learning, in addition to the main topics.

Acknowledgements

I would like to thank the many researchers I have been in contact with during the semester – George Forman, Eui-Hong (Sam) Han, Jason Rennie, Thorsten Joachims and Tom Mitchell – for taking the time to explain the details of their experiments.

I would also like to thank my wife Sheila for much patience and proof reading during the last months. Thanks also to my second proof reader Terese.

Last but not least, I would like to thank my main supervisor Ph.D. Student Robert Neumayer for making it possible for me to conduct this work while living in Oslo. Albeit many long hours, it has been a pleasure working with you. Many thanks also to Professor Kjetil Nørvåg for suggesting this problem for my Master's Thesis.

Oslo, June 11, 2009

Øystein Løhre Garnes

Summary

Text categorization is the task of discovering the category or class text documents belongs to, or in other words spotting the correct topic for text documents. While there today exists many machine learning schemes for building automatic classifiers, these are typically resource demanding and do not always achieve the best results when given the whole contents of the documents. A popular solution to these problems is called feature selection. The features (e.g. terms) in a document collection are given weights based on a simple scheme, and then ranked by these weights. Next, each document is represented using only the top ranked features, typically only a few percent of the features. The classifier is then built in considerably less time, and might even improve accuracy.

In situations where the documents can belong to one of a series of categories, one can either build a multi-class classifier and use one feature set for all categories, or one can split the problem into a series of binary categorization tasks (deciding if documents belong to a category or *not*) and create one ranked feature subset for each category/classifier.

Many feature selection metrics have been suggested over the last decades, including supervised methods that make use of a manually pre-categorized set of training documents, and unsupervised methods that need only training documents of the same type or collection that is to be categorized. While many of these look promising, there has been a lack of large-scale comparison experiments. Also, several methods have been proposed the last two years. Moreover, most evaluations are conducted on a set of binary tasks instead of a multi-class task as this often gives better results, although multi-class categorization with a joint feature set often is used in operational environments.

In this report, we present results from the comparison of 16 feature selection methods (in addition to random selection) using various feature set sizes. Of these, 5 were unsupervised , and 11 were supervised. All methods are tested on both a Naïve Bayes (NB) classifier and a Support Vector Machine (SVM) classifier. We conducted multi-class experiments using a collection with 20 non-overlapping categories, and each feature selection method produced feature sets common for all the categories. We also combined feature selection methods and evaluated their joint efforts.

We found that the classical supervised methods had the best performance, including Chi Square, Information Gain and Mutual Information. The Chi Square variant GSS coefficient was also among the top performers. Odds Ratio showed excellent performance for NB, but not for SVM. The three unsupervised methods Collection Frequency, Collection Frequency Inverse Document Frequency and Term Frequency Document Frequency all showed performances close to the best group. The Bi-Normal Separation metric produced excellent results for the smallest feature subsets. The weirdness factor performed several times better than random selection, but was not among the top performing group. Some combination experiments achieved better results than each method alone, but the majority did not. The top performers Chi square and GSS coefficient classified more documents when used together than alone. Four of the five combinations that showed increase in performance included the BNS metric.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

This chapter provides a short introduction to the topics of the project. We first present the motivation behind the assignment and define the problem and goals. Next, we define the scope of the work, and finally we list the outline of the report.

1.1 Motivation

Categorizing text documents means to discover their category or topic from a set of predefined categories, e.g. 'sports' or 'economics'. Text categorization is an important field within natural language processing. Its application areas are many and the need for them is increasingly important as the amounts of information continue to grow. Junk mail filtering has been an important area for text categorization the last decade, as have portals with hierarchies of web sites, digital libraries and more. But the general task of filing a text document in the correct location – or spotting its correct topic – will exist as long as digital written texts are being produced. Other examples include publishing newspaper articles in the correct category or storing a digital document correctly in an archive or library.

Automatic text categorization was first done as early as the sixties, though the lack of computer power made it infeasible for a long time. During the last decade or so however, we have seen a lot of efforts in the area. While computers today are capable of learning and performing text categorization within reasonable time limits, growing amounts of data makes TC challenging today as well.

When classifying text documents one considers the *features* of a document, typically these correspond to terms. Not all features are equally helpful for deciding which category a documents belongs to. One can say that they convey less information, while some features may even be regarded as noise. Selecting a (good) subset of these features has emerged as a research field itself, named **feature selection**. Selecting a subset of features can give both huge savings in computation time and increase in accuracy.

Many methods for ranking and selecting features have been presented, and the main task of this assignment is to compare promising methods in the same system. Feature selection methods have been compared before, but the number

of methods compared in each paper is often sparse. Also, several methods have been presented as late as 2008, and there is a so far unfulfilled need to compare these against each other and the classic methods.

The following sentences about feature selection methods from Sebatiani's much appreciated paper [Seb02] provides good motivation:

> [...] it should be noted that these results are just indicative, and that more general statements on the relative merits of these functions could be made only as a result of comparative experiments performed in thoroughly controlled conditions and on a variety of different situations (e.g., different classifiers, different initial corpora, ...).

Such comparative experiments are exactly what this thesis will include.

1.1.1 Problem definition and goals

The short version of the assignment text reads:

> Information retrieval and text mining methods operate on the terms found in text documents. As such, every term found in a collection is analyzed and used for further processing. The process of feature selection is performed in order to reduce the number of terms to be used in further analysis (i.e. to identify the most important terms beforehand). The task of this project is to compare a range of feature selection techniques with the goal of a thorough performance evaluation.

The main goal of the assignment is evidently to compare several feature selection techniques. Moreover, the performance evaluation should be thorough – e.g. include several performance metrics suitable for different situations.

We can extend the goals to include the following:

1. Comparing various feature selection methods on more than one classifier, e.g. a Naive Bayes classifier and a Support Vector Machine.

2. Comparing both supervised (using the category information in a pre-classified training set) and unsupervised selection methods.

3. Comparing several recently porposed methods that have not yet been parts of large-scale testing.

4. As there are so many variable parameters, performance results from one researcher cannot be compared to results from another, even if the same collection and classifier is used. Hence, comparing as many feature selection methods as possible in one system is a subgoal.

1.1.2 Project Scope

This thesis will include a preliminary study of the many aspects of text categorization in general and feature selection in particular. It will cover many feature selection methods, but a complete comparison would be infeasible in the time slot given.

A prototype for distributed text categorization based on Apache Lucene[1] is in development. This project work will include importing and indexing well-known text benchmark collections, implementing feature selection techniques and computing feature rankings in the prototype. The selected features will be exported to machine learning suites with the documents represented using those features, and classification models will be generated on powerful computers.

We have decided to make use of the well-known data mining suite Weka[2] for building the classifiers. Within Weka, we have decided to focus on the classic Naive Bayes learner, and a Support Vector Machine learner which has seen increasing popularity for text categorization the last decade.

Both Chapters 2 and 3 will narrow down the scope of the project, as there are many details to decide on even though the assignment may sound rather straight forward.

1.1.3 Report outline

The rest of the report will be structured as follows:

Chapter 2 – Text Categorization will introduce text categorization, as well as explain which choices we have made in our work.

Chapter 3 – Feature Selection will explain the details of feature selection. A range of state-of-the-art methods will be presented, using a junk mail filtering example.

Chapter 4 – Experiments and Results will present our experiments including the text collections used. Also, the performance results will be presented and discussed.

Chapter 5 – Conclusion and Outlook will provide some concluding remarks of the results, and suggestions for future work.

[1] `http://lucene.apache.org/java/docs/`
[2] `http://www.cs.waikato.ac.nz/ml/weka/`

Chapter 2

Text Categorization

This chapter will introduce the concept of text categorization, also known as text classification or topic spotting. We will present different types of text categorization, popular applications, and components of a text categorization system.

We will also touch upon feature selection, an important concept within text categorization which is the main topic of Chapter 3 and this report in general.

Introduction

Text categorization is the task of discovering the category or categories that a text document belongs to, from a fixed set of predefined categories. In other words to assign category labels to documents. As mentioned, it is also called topic spotting, as it can be seen as the problem of spotting the topic of text documents. Text categorization can of course be done manually, but in this report we look at automated text categorization systems.

Other meanings of text categorization/classification As explained in the excellent paper [Seb02], somewhat different activities are referred to as automated text classification. In this report, we mean the automatic assignment of category labels to documents, where the categories available are known in advance. Others mean the dicovery of the categories, or the discovery process and the assignment of category labels – more often known as clustering. An even wider definition of the word includes any process of grouping documents by categories, that is, both the above mentioned variations.

2.1 Types of Text Categorization

Text categorization can be divided into sub-types in various ways. In this section, we list some of them, and discuss what they are good for.

2.1.1 One or more Categories for a Document

Text categorization (TC) systems can either have several categories to choose from, or just two (e.g. interesting or uninteresting). Also, they can label each

document with either exactly one, or several (0-k) category labels. We can break down the various types as follows:

Multiple Label TC Some systems and applications can assign from none to multiple category labels (zero, one, several, or even all) to each document. Articles in an online news paper could for instance belong to both the 'international' category and the 'sports' category. This type of text categorization systems are also called overlapping categories, as mentioned in [Seb02, Page 3].

Single Label TC Single label systems assign exactly one category label to each document. There can be many categories to choose from, however. Our main example, presented in Section 3.5 is a single label text categorization problem, with three possible categories. Likewise, the experiments presented in Chapter 4 are single label.

Binary TC Binary text categorization is a special case of single label text categorization, but here only two categories are available. Moreover, each document *has* to be labeled with one of these categories. Posts in a news feed could for instance be labeled interesting or uninteresting for a user. A junk mail filter is another example where a binary text categorization system could be applied.

Note that binary classification also is important because it is often used as a subroutine in many multi-class (i.e. multiple label and single label) tasks, see [For07].

2.1.2 Knowledge Engineering vs. Machine Learning

Up until the nineties, text categorization was often done the *knowledge engineering* way [Seb02]. Expert systems were created by knowledge engineers and domain experts. These systems consisted of rules, that contained enough knowledge to classify the documents.

In the nineties, the machine learning approach gained popularity. In this approach, artificial intelligence is used to create classifiers that can assign category labels to documents automatically. The classifiers are trained on manually pre-classified documents called training documents. This approach is preferable because of faster re-training when the categories or other aspects change. Because of limited computing power however, it was not feasible before the late eighties.

This report concerns the machine learning approach to text categorization, using a Naïve Bayes classifier and a Support Vector Machine classifier.

2.1.3 Hard Categorization vs. Ranking TC

As explained in Section 2.1.2, text categorization can be done by expert systems concisting of rules created by knowledge engineers. An automated text categorization system using machine learning can function as an aid for the knowledge engineers when creating rules, or for anyone performing manual text categorization, as they only need to consider the top ranked categories for each document (or the top ranked documents for each category). This is called *ranking text categorization* [Seb02].

On the other hand, systems using the machine learning approach only, are called hard text categorization systems. Hence, this report concerns hard text categorization.

2.1.4 Document Pivoted vs. Category Pivoted TC

Text categorization can be performed from different 'perspectives': Either one can find all categories that a document belongs to (called document pivoted text categorization), or one can find all documents belonging to each category in turn (called category pivoted text categorization). The documents should be labeled with the same category label(s) independent of which method is used, but they have different application nevertheless:

Document pivoted text categorization is suitable in situations where new documents arrive sequentially and the categories are stable, for instance in a junk mail system. Category pivoted text categorization is suitable when there is a given set of documents that should be categorized, and when new categories are added to an existing system.

The feature selection methods used in this report all allow the construction of classifiers (see Section 2.4.3) capable of working both in a category pivoted and a document pivoted manner.

2.1.5 Supervised vs. Unsupervised Text categorization

Automatic text categorization can either be supervised or unsupervised. The unsupervised type (e.g. text clustering) both creates and assigns category labels solely by observing the documents that are to be clustered/categorized.

In this report however, we look at supervised learning. Here, a set of (sometimes pre-classified) documents must be available (see Section 2.4.4). Also, a predefined set of categories must be given. The learning process is 'supervised' or guided by the known categories and the training documents. Notice however, that we compare both supervised and unsupervised *feature selection* methods, where the former uses the category information in the training data and the latter does not.

2.1.6 Exogenous and Endogenous Knowledge

Text categorizing can utilize exogenous knowledge, i.e. data from an external source, to aid the decision process. Meta-data like author, keywords, date of publication etc. could for instance be exploited. The document content itself by comparison, contains endogenous knowledge [Seb02, Section 2.1].

We are in this report concerned with feature selection and text categorization methods that use endogenous knowledge only.

2.2 Applications

Text categorization is a basic building block in many information systems. It has probably been carried out for almost as long as written languages have existed, although automatic text categorization only has been around the last half decade.

In this section, we will look at some of the areas where text categorization is used (manually or automatic).

Automatic Indexing for Controlled Vocabulary IR System A classic text categorization application is called *text indexing*. Here, documents in a boolean IR system are labeled with one or more keywords (or categories). The IR system can then return documents matching a user's keyword-based query [FS07, Chapter IV].

Notice that the keywords are defined in a controlled vocabulary, e.g. a thesaurus like the MESH thesaurus for medicine.

Document Organizing The organization or sorting of documents into 'bins' or categories is a common application. For instance, newspapers may want to automate the sorting of classified ads into e.g. 'personal', 'real estate' etc., or file their articles into 'national', 'international', 'sports' and other categories. For large newspapers, doing this manually could be time-consuming. Whether to use single or multiple category labeling[1] depends on the specific organizing job.

Text Filtering Text filtering here refers to the task of categorizing documents that arrive to the classifier one by one [For07]. Examples include junk mail filters, where the binary type of text categorizing typically is used (e.g. a message is either marked as spam or not spam). As discussed in [FS07, Chapter IV], an important question in such systems is the weighting of either recall errors (a document missing from a category, e.g. 'not spam') or precision errors (a document filed under a category where it does not belong). For the mentioned spam filter, recall errors are typically seen as more important to avoid than precision errors (e.g. one would rather receive a few spam messages than miss one important message).

Also, any service where a user can create a profile specifying her interests in order to filter some sort of message or document stream (e-mail, advertisements, news articles, event notifications, scientific papers, etc.) could make use of automatic text categorization. Indeed, the example used in this report (see Section 3.5) is of this type. In such systems, single or multiple label text categorization is appropriate.

Computational Linguistics Computational linguistics is the use of computers for processing natural language grammars and linguistics. Several subareas of computational linguistics can be tackled by automatic text categorization, including *word sense disambiguation*, *context-sensitive spelling correction*, *prepositional phrase attachment*, *part-of-speech (POS) tagging*, and *word choice selection* [Seb02, Section 3.4].

Word sense disambiguation (the task of deciding which of a word's several meanings/senses is meant in a sentence or context) for instance, can be seen as a text categorizing task if we let the word senses act as categories, and the word occurrence contexts act as documents.

[1] See section 2.1

Hierarchical Categorization of Web Pages The World Wide Web consists of millions of web sites, and for many years already, Internet portals (such as the Yahoo! Directory[2] [FS07, Chapter IV]) have been popular starting points for navigating to relevant web sites. These portals allow a user to browse through a hierarchical structure of categories, narrowing the scope of possibly relevant sites for each click. Furthermore, the user may often either make direct use of hyperlinks to a desired web site directly, or he or she may choose to submit a query within the selected category.

The number of web sites continues to grow, and manual web site categorization of a noticeable size has for a long time been infeasible. The text categorizing system could be built up assigning one classifier at each node in the hierarchy [Seb02, Section 3.5], and it should allow new categories to be added and obsolete ones to be removed. A minimum and maximum number of documents for each category could for instance be applied.

2.3 Document Representation

In this section, we present some information retrieval basics needed to grasp the technical details of text categorization and the feature selection methods.

2.3.1 The Vector Space Model

The vector space model is one of the most popular models used in Information Retrieval [MRS08, BYRN99] and Text Categorization. It is a basic building block in many operations, including document ranking in search engines, document clustering and categorization. Since it is a fundamental part of the systems used in this report, a short introduction to the model follows.

In the vector model, each document can be seen as a vector of terms, where each term is assigned a weight based on some weighting scheme. The most basic weighting scheme uses binary weights ($w = 1$ if the term is present in the document, and $w = 0$ if not). A better and much referenced weighting scheme is the tf-idf scheme. Here, a term's frequency (how often it appears in a document) is multiplied by its inverse document frequency (the total number of documents divided by the number of documents containing the term). See Section 2.3.2 for more on this.

In a search engine, each query is typically represented by the same kind of vector in the same vector space as the documents, enabling fast similarity comparisons. In text categorization, all documents in the training-, test- and validation sets (see Section 2.4.4) and the documents to be classified are indexed and represented by the same model, typically the vector space model.

2.3.2 Weighting Schemes

In Information Retrieval applications, each term or feature in a document is typically given a feature *weight* based on a weighthing scheme. This section briefly lists some basic ideas, including the tf-idf weighting scheme that we use in our experiments.

[2] `http://dir.yahoo.com/`

Term Frequency How often a term i appears (called the *term frequency (tf)*) in a document j ($tf_{i,j}$) and corpus (tf_i) is important, as it carries knowledge of its relevance. However, some calculations are typically done to make it more useful. It can be *normalized* [BYRN99] by incorporating the maximum frequency over all terms in the document ($max_l freq_{l,j}$):

$$ntf_{i,j} = \frac{tf_{i,j}}{max_l freq_{l,j}}$$

Another way of refining the term frequency is to make it *relative* [Cim06] to the sum of all other term frequencies (tf_k), either for the whole collection:

$$rtf_i = \frac{tf_i}{\sum_k tf_k}$$

or per document j:

$$rtf_{i,j} = \frac{tf_{i,j}}{\sum_k tf_{k,j}}$$

Document Frequency Further on, the notion of *document frequency* can defined as the number of documents j in a corpus D where the term i occurs:

$$df_i = |\{d \in D \,|\, d \; contains \; i\}|$$

Inverse Document Frequency One can also consider the *inverse* document frequency (or *idf-factor*). This measure will favor terms that appear in few documents, and penalize those that appear in many documents:

$$idf_i = \log \frac{|D|}{df_i}$$

Tf-idf Weighting When combined, the term frequency and the inverse document frequency can be used as a weight of a term, typically done in the vector-model in IR:

$$tf\text{-}idf_{i,j} = tf_{i,j} \cdot idf_i$$

The tf-idf can be computed in various ways [BYRN99, Cim06], and the above shown is the most basic. We employ tf-idf weights in our experiments presented in chapter 4.

2.4 Components of a Text Categorization System

In this section, we briefly explain the common components of a text categorization system, that is, the operations necessary or usually executed when performing text categorization.

2.4.1 Preprocessing

Before the documents can be categorized, they are often preprocessed, including sentence and term splitting, word stemming, etc. What is done in this phase depends on the system and application. In this section, we present some possible preprocessing tasks.

Tokenization Tokenization takes a text (i.e. a string) and discovers sentences and 'tokens' [FS07, Chapter III]. This may seem like a 'no-brainer' (just split the string at every space), but this is not the case. Abbreviations like 'etc.', 'i.e.' and 'e.g.' are typically written with punctuations, so punctuations do not always end sentences. Also, hyphens, digits, cases and more need to be handled. Combinations of rules and predefined lists can be used to decide what to do with ambiguities.

How tokens are *represented* is different from system to system. Some include characters only, while others include position in the text, token *type* etc. We used Lucene's StandardTokenizer in our experiments.

Stemming Stemming reduces a word to its stem [BYRN99]. For instance the word *connection* can be reduced to its word stem *connect*. The same goes for the following variations of the same word stem: *connected*, *connecting*, and *connections*. A famous algorithm for stemming is the Porter stemmer [Por80] that is based on removing suffixes from words (e.g. removing s from plural words). We employ Lucene's Snowball Analyzer, which includes a stemmer based on the Porter algorithm.

Note that stemming usually reduces the number of features in the corpus, since several words typically are reduced to the same word stem. Reducing the number of words/features is discussed in the next section.

2.4.2 Term Space Reduction

Before the machine learning starts, the term space is often reduced (that is, a subset of the available features are selected), to save computing time or even increase accuracy. Term Space reduction is explained in detail in section 3.1, including Feature Selection, Extraction or Construction.

Removal of Stop Words Stop words, also called function words or common words, are words that are so common in a text collection that they do not add any information to the categorization process. For instance, the words 'the', 'it', 'and' etc., are probably found in most English text documents. For the classifier however, such words are just as computationally demanding as any other words, so it is very common to remove them. Note that stop words are a subjective matter, there is no official definition of which words are stop words or not.

We performed individual stop word removal in addition to Lucene's standard list of English stop words.

2.4.3 Machine Learning/Induction

The machine learning component is the most important part of an automatic text categorizing system. The classifier is where the actual decisions are made for which topic(s) a document should be labeled with. A classifier is created in an inductive step called a learner, where the contents (the features) of a set of pre-classified training documents are used to 'teach' the classifier which features new, unseen documents should contain to be labeled as belonging to that category.

Several machine learning techniques can be used for text categorizing, including probabilistic classifiers (e.g. Naïve Bayes), decision tree classifiers (e.g.

the ID3 algorithm [Qui86]), neural networks, and more [FS07, Chapter IV]. We build and evaluate Naïve Bayes and Support Vector Machine classifiers in our experiments.

2.4.4 Training, Test, and Validation Sets

When using supervised feature selection methods (as those presented in Section 3.7), the handling of the pre-classified documents (the traing data) is critical.

A common practice is to split the training data into two sets; a training set Tr and a test set Te [Seb02, Section 4.1]. The training set Tr is used to train the classifier (for instance using machine learning with features selected by the methods from Section 3.7). The test set Te is then used to test/evaluate how effective the classifier turned out to be. This test will thus evaluate the joint quality of the training data, the feature selection method, and the machine learning algorithm used in the classifier. This way of using the training data is called the *train-and-test* approach.

The classifier may some times be tuned with internal parameters, and to be able to validate which settings produce the best result, a validation set Va is introduced. One performs several tests of the qualifier, and uses the validation set to land on the best settings. These sets must be kept completely separate from each other, to avoid misleadingly good results.

Another way to evaluate the classifier is called the *k-fold cross-validation* approach [Seb02] or *hold-out testing* [Mla98]. Here, the pre-classified documents are first randomly split into k disjoint test sets Te_k (with the rest of the documents as training and validation sets). Then, the *train-and-test* approach is applied, creating k classifiers. In the end, the average result of the k classifiers should be more reliable than the results from one single *train-and-test* run. While cross-validation is recommended by many (e.g. [For07]), it demands more from the text categorizing software and computers, and seems not to have been used in the majority of experiments we have studied. It is employed in [YH08], [ZH07] and [Mla98]. We used 10-fold cross-validation using Weka in the experiments presented in Chapter 4.

If the classifier is to be used for actual operational categorizing purposes (and not just evaluating and comparing it to other classifiers as we do), one should consider using *all* pre-classified documents for training if the classifier, after evaluation is done. Training the classifier on the whole set of pre-classified documents should result in higher effectiveness than partial training, and the results from the *train-and-test* or the *k-fold cross-validation* evaluation approaches should be treated as minimum effectiveness estimates.

Splitting the training data in time periods Some [BMGMF08, RS99, LR94] split the pre-classified documents in *time periods*, that is, the oldest documents in the training data are used for training, the newer ones are used for testing. This can be an advantage, as it imitates a real life situation. If we were to create an operational text categorization classifier, the only training data available would be that which has been produced up until now. Likewise, we would typically use the classifier to categorize documents being produced in the future. Not all categorizing tasks are for future data, however. Also, this method requires a date label on the pre-classified documents. We do not

employ such a split in the experiments presented in this report, as we use 10-fold cross-validation.

2.5 Evaluation

Automatic text categorization systems are evaluated based on how well they perform compared to the *true* category information of documents. Using documents with known category information as described in Section 2.4.4, we can evaluate a classifiers performance in various metrics:

The **percentage** of correctly categorized documents is a natural starting point for measuring the performance of a classifier. However, as some important information may be hidden behind this metric, a few other measures are commonly used as well. Before presenting these, we show some important document counts used in the computations.

	Document actually belongs to category	Document actually does not belong to category
Classifier says document belongs to category	True positives (TP)	False Positives (FP)
Classifier says document does not belong to category	False negatives (FN)	True negatives (TN)

Table 2.1: Notation: Contingency table for evaluation measures

Here, a true positive (TP) is a *correctly* assigned positive class label, while a false positive (FP) is an *incorrectly* assigned positive class label. A false negative (FN) is a document that actually belongs to the category, but was not recognized as such by the classifier. A true negative (TN) is a document not belonging to the class, and the classifier correctly did not place in the class.

Hence, the sum of true and false positives means the *actual* number of documents that the classifier placed in that class, while the sum of true positives and false negatives means the number of documents that actually belong to a category, no matter what the classifier says.

The **precision** of a classifier is defined as the the number of true positives divided by the number of true *and* false positives. The **recall** of a classifier is defined as the the number of true positives divided by the number of true positives and false *negatives* [BYRN99].

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

The precision level is sometimes referred to as a level of exactness, while the recall level measures completeness. Some applications are more concerned with one than the other. For instance, when filtering junk e-mail from important e-mail, we would typically rather accept occasional spam messages in the inbox than important messages in the junk mail folder. Hence, when classifying spam

we would like a high degree of exactness or precision, while when classifying relevant messages, we would like a high degree of completeness or recall.

The precision ad recall levels are often combined into a single metric called the **$\mathbf{F_1}$-measure** (or sometimes just F-measure). The number one says that precision and recall are equally weighted.

$$F_1 = \frac{2(precision \cdot recall)}{precision + recall}$$

The **$\mathbf{F_1}$-measure** is in fact a special case of the **$\mathbf{F}_\beta$-measure**:

$$F_\beta = \frac{(1 + \beta^2)(precision \cdot recall)}{\beta^2 \cdot precision + recall}$$

where β specifies how many times more recall should be weighted over precision [vR79]. For instance, the F_2-measure weights recall twice as much as precision, while the $F_{0.5}$-measure weights precision twice as much as recall. In our experiments, we report the percentage of correctly categorized documents, precision, recall, and F_1-measure of the classifiers performances.

Moreover, when categorizing documents into multiple classes, the precision, recall and F-measures can be reported in both a macroaverage and microaverage metric. **Macroaverage** means the arithmetic mean over all classes, while **microaverage** means the average weighted by the class distribution. The macroaverage weights each problem equally, while the microaverage weights each document classification equally. Hence, for highly skewed class sizes, these figures may show quite different values. In our experiments however, we have used a corpus with rather similar document counts for each class, so micro- and macroaverage values are rather similar. Thus we present only the macroaveraged precision, recall and F_1-measure.

2.6 Text Categorization Summary

Here we sum up the choices we have made, before looking at feature selection in the next chapter:

This report focuses on automated, single label text categorization systems, where categorization means the automatic assignment of category labels to documents, where the categories available are known in advance. It concerns the machine learning approach to text categorization, and not the knowledge engineering approach. We consider *hard* text categorization, and we make use of *supervised* machine learning methods for text categorization. Notice however, that we compare both supervised and unsupervised *feature selection* methods. We are concerned with feature selection and text categorization methods that use endogenous knowledge only. This report concerns systems based on the vector model. We use binary weights in the examples, but tfidf weights in the experiments presented in Chapter 4. These experiments includes tokenization by Lucene's StandardTokenizer and stemming by Lucene's SnowballStemmer. Our experiments make use of 10-fold cross-validation instead of pre-defined training and test sets.

Chapter 3

Feature Selection

In this chapter we will introduce the concept of 'features', why we select some features over others, and how it's done. Then we will present a practical notation, and a feature selection example. This example will be used throughout the chapter, when we go through some of the important feature selection techniques that exist.

Introduction

Feature Selection is the step of selecting[1] some features (e.g. words or terms) to be used when building an automatic classifier for text categorization. Instead of representing a document with all its features, we can represent it using only the selected ones. This way, the classifier has to handle less data.

As pointed out in [For07] there are two main reasons for selecting some features over others:

Accuracy Firstly, studies have shown that machine learning algorithms can produce better results when *not* considering all the features. It would be reasonable to think that the more features considered, the more accurate the classifier would become. However, some features do not add more information (they are merely noise), and removing these can make the classifier perform more accurately.

Scalability Secondly, as machine learning algorithms are resource demanding (computation power, memory need, network bandwith, storage, etc.), running them on a subset of the features typically yields significant time savings. The ability to work with a small subset of the features also ensures scalability. [Mla98] reported good accuracy even with subsets of just 2% of the available features.

Combining these reasons, we can say that feature selection is the task of selecting the subset of features with the best signal-to-noise ratio.

In this report, we select features by ranking them according to some measure, and then use the i best features for creating the classifier. This kind of feature selection is sometimes called 'feature selection by feature ranking', e.g. in [MK07].

[1] Note that there are other ways of creating a feature set than *selection*, as explained in Section 3.1

Aggressiveness Feature Selection can be modest or aggressive. The fewer features selected, the more aggressive the feature selection can be said to be. Feature selection can be said to have a high reduction factor, i.e. the number of available features are highly reduced. A high reduction factor is the same as an aggressive feature selection process.

So what is a feature? A feature is some characteristic, detail or aspect of something. In a text document, the words or terms make up the most obvious features, but as we shall see in the next section, other ways of finding features exist as well.

For the task of classifying documents, we need good quality features. Good quality features contain much information that the classifier can use to decide which category a document belongs to. Hence, a word that occurs in all documents in one category in the training set and in none of the other categories, would probably be a good quality feature for that category. Poor quality features contain less information about the class membership. For instance, stop words (like 'the', 'it', 'and' etc.) will probably occur in all categories, and would not help the classifier much in the decision process.

Poor quality features are also called irrelevant features [LM07b], and the performance of the classifier is maintained (or even raised) if they are removed. A special kind of irrelevant features are the redundant features. They are useful for the classifier themselves, but can be removed since there are other features contributing the same information. Note that by removing redundant features, the classifier performance remains the same, while the computation time falls.

Some features can seem like good features in the training data, but then turn out not to work well in real life. Such features are called *noise features*, and when such features are selected by the feature selection technique, it is called *overfitting*, i.e. the classifier trained with the selected features (including the noise features) will be very good at categorizing the training documents, but not so good for other documents.

We employ Lucene's StandardTokenizer in our experiments. From the Lucene API we can see that StandardTokenizer:

- Splits words at punctuation characters, removing punctuation. However, a dot that's not followed by whitespace is considered part of a token.
- Splits words at hyphens, unless there's a number in the token, in which case the whole token is interpreted as a product number and is not split.
- Recognizes email addresses and Internet hostnames as one token.

In practice, this means that StandardTokenizer recognizes e-mail addresses, time and dates, and these becomes our features along with other words and terms.

To give the reader an idea of what features are, we list some features from the 20 Newsgroups collection, and the feature weights given to them by the feature selection method called Information Gain:

- 18-year 7.420006103018295E-4
- brash 7.420006103018295E-4
- skirmish 7.420006103018295E-4

- 99.9 7.419250950011502E-4
- typist 7.415805449717539E-4
- bard 7.414468753861314E-4
- mccartney 7.414468753861314E-4
- 51-day 7.410826647324953E-4
- turmoil@halcyon.com 7.410826647324953E-4
- whacko 7.410826647324953E-4

3.1 Feature Selection vs. Construction and Extraction

As explained in the introduction to this chapter, feature selection can be a crucial step in a text categorizing system. It is often infeasible to use the whole document content in the machine learning induction process as it will take too long to finish.

The methods used to reduce the size of a document's vector representation are sometimes called *dimensionality reduction* methods [CHTQ09], [YH08], [Seb02, Page 13], [LJ98] and [YP97]. There are several ways of reducing this dimensionality:

In feature *selection*, we choose a set of features to keep, and discard the rest. This reduces the number of single features, and is thus sometimes called *feature space reduction* or *term space reduction* (TSR) [Seb02, Page 14]. The document vectors explained in the vector space model in Section 2.3.1 is reduced, and the machine learning algorithms receive less data to handle. This is the kind of dimensionality reduction we evaluate in this report.

Feature Construction and Feature Extraction are methods that transform the set of features available into a new (typically smaller) set of features. They are collectively referred to as Feature Transformation methods. Feature Construction methods transform the features by finding relationships between existing features, for instance using the sum of two features as a new feature (a *phrase*). Feature Extraction methods create new features that typically do not resemble the original features at all. One example here is Latent Semantic Indexing (LSI). We will not be concerned with feature transformation methods in this report.

This report concerns feature *selection* only. While we most often refer to a feature selection *method*, other common terms are metric, function, technique, equation, and algorithm. Feature selection methods are sometimes referred to as TSR functions, feature evaluation functions (FEF), feature ranking, scoring or filtering methods etc.

3.2 Local vs. Global Feature Selection

Feature selection methods typically create a ranked list of features that are (hopefully) good features to train a classifier for one specific category. That is, feature selection methods are usually category specific, and the classifier that uses the selected features is usually trained to decide if documents belong to that specific category.

To handle one set of features for each category can be impractical however, so one sometimes wishes to merge the ranked lists (or the selected features) into one common feature list for use with all the categories/classifiers. Such a common list is called a 'global' or 'category-independent' feature list [Seb02]. It should be noted however, that the performance of text categorizing typically is somewhat lower with global feature selection, as the features are less *specialized* for the categories.

This report concerns global feature selection, i.e. we employ *one* common (joint) ranked feature list for all classifiers.

3.3 Filter vs. Wrapper approach

Feature Selection methods are often grouped into *filters* and *wrappers*. Filter methods measure feature relevance by applying statistical tests to the feature counts and/or the class labels of the training set. Wrapper methods measure feature subset usefulness by using AI search methods like greedy hill-climbing or simulated-annealing to search the space of all feature subsets via cross-validation with the same induction algorithm that is later used for building the classifier [For07, Seb02]. Where filter methods evaluate each feature independently, wrappers evaluate feature sets as a whole, which in theory would avoid redundant features and lead to better results. However, wrapper methods are computationally infeasible for large datasets, and are also more prone to overfitting, so filter methods are more commonly used.

This report concerns the filter approach to feature selection only.

3.4 Notation

When presenting the different methods for feature selection, we use a compact notation listed below and depicted in Table 3.1. All the Nvariables below are document counts from the training set of the document corpus at hand.

The F is for Feature, and the features are defined in the introduction to this chapter. Categories C_k are labels or topics of documents, as defined in Chapter 2.

N is the total number of documents in the training set.
N_{C_k} is the number of documents in category C_k.
$N_{\overline{C_k}}$ is the number of documents not in category C_k.
N_F is the number of documents containing feature F.
$N_{\overline{F}}$ is the number of documents not containing feature F.
N_{F,C_k} is the number of documents containing feature F in category C_k.
$N_{\overline{F},C_k}$ is the number of documents not containing feature F in category C_k.
$N_{F,\overline{C_k}}$ is the number of documents containing feature F not in category C_k.
$N_{\overline{F},\overline{C_k}}$ is the number of documents not containing feature F not in category C_k.

The document counts are used for estimating probabilities by most supervised feature selection algorithms. Thus a coherent representation should make it intuitive for the reader to follow the algorithms and compare them.

	$v_{category} = 1$	$v_{category} = 0$	
$v_{feature} = 1$	$N_{F,C_k} = 351$	$N_{F,\overline{C_k}} = 102$	$N_F = 453$
$v_{feature} = 0$	$N_{\overline{F},C_k} = 0$	$N_{\overline{F},\overline{C_k}} = 210$	$N_{\overline{F}} = 210$
	$N_{C_k} = 351$	$N_{\overline{C_k}} = 312$	$N = 663$

Table 3.1: The table shows a feature-category contingency table example, where the values are document counts. $v_{feature} = 1$ means that the feature in question is present in a document, while $v_{feature} = 0$ means it is not present. Similarly, $v_{category} = 1$ means that a document belongs to the category, and $v_{category} = 0$ means it does not belong to the category. The four permutations of $v_{feature}$ and $v_{category}$ make up the four counts N_{F,C_k}, $N_{F,\overline{C_k}}$, $N_{\overline{F},C_k}$, and $N_{\overline{F},\overline{C_k}}$. The values shown here are fictive. Notice that the values ($v_{feature}$ and $v_{category}$) in each row and column are summed up in the far right column and the lower row, respectively, creating N_F, $N_{\overline{F}}$, N_{C_k}, $N_{\overline{C_k}}$ and the overall total N.

3.5 Example

Throughout the report, we will use an e-mail filtering example, as shown in Table 3.2. The example is a single label text categorization problem with multiple categories, as explained in Section 2.1.

The messages (documents) in this example are very simplified compared to real life documents, even if one imagines that they have had their stop words removed. Our messages consist of a very small set of words, indeed, message #9 consists of just the word 'ski'. The example is created this way to show the differences of the feature selection methods, rather than to resemble real life documents.

We will in this example use binary weights for clarity. The e-mail example can also be presented using contingency tables, as in Tables 3.3, 3.4 and 3.5. These tables show document counts for (feature, category) pairs for the spam, business, and private categories, respectively.

	Number of occurrences for the available features					
Message	**'viagra'**	**'save'**	**'erection'**	**'cell'**	**'ski'**	**class C_k**
#1	1	0	1	0	0	Spam
#2	1	1	1	0	0	Spam
#3	2	0	1	0	0	Spam
#4	1	0	0	0	0	Spam
#5	0	1	0	0	1	Business
#6	0	1	0	1	0	Business
#7	0	0	0	3	0	Business
#8	1	0	0	0	1	Private
#9	0	0	0	0	1	Private

Table 3.2: E-mail filtering example. The table shows the number of times a feature occurs in a document.

	$v_c = v_{spam} = 1$	$v_c = v_{spam} = 0$	
$v_f = v_{viagra} = 1$	$N_{F,C_k} = 4$	$N_{F,\overline{C_k}} = 1$	$N_F = 5$
$v_f = v_{viagra} = 0$	$N_{\overline{F},C_k} = 0$	$N_{\overline{F},\overline{C_k}} = 4$	$N_{\overline{F}} = 4$
	$N_{C_k} = 4$	$N_{\overline{C_k}} = 5$	$N = 9$

(a) feature 'viagra' and category 'spam'

	$v_c = v_{spam} = 1$	$v_c = v_{spam} = 0$	
$v_f = v_{save} = 1$	$N_{F,C_k} = 1$	$N_{F,\overline{C_k}} = 2$	$N_F = 3$
$v_f = v_{save} = 0$	$N_{\overline{F},C_k} = 3$	$N_{\overline{F},\overline{C_k}} = 3$	$N_{\overline{F}} = 6$
	$N_{C_k} = 4$	$N_{\overline{C_k}} = 5$	$N = 9$

(b) feature 'save' and category 'spam'

	$v_c = v_{spam} = 1$	$v_c = v_{spam} = 0$	
$v_f = v_{erection} = 1$	$N_{F,C_k} = 3$	$N_{F,\overline{C_k}} = 0$	$N_F = 3$
$v_f = v_{erection} = 0$	$N_{\overline{F},C_k} = 1$	$N_{\overline{F},\overline{C_k}} = 5$	$N_{\overline{F}} = 6$
	$N_{C_k} = 4$	$N_{\overline{C_k}} = 5$	$N = 9$

(c) feature 'erection' and category 'spam'

	$v_c = v_{spam} = 1$	$v_c = v_{spam} = 0$	
$v_f = v_{cell} = 1$	$N_{F,C_k} = 0$	$N_{F,\overline{C_k}} = 2$	$N_F = 2$
$v_f = v_{cell} = 0$	$N_{\overline{F},C_k} = 4$	$N_{\overline{F},\overline{C_k}} = 3$	$N_{\overline{F}} = 7$
	$N_{C_k} = 4$	$N_{\overline{C_k}} = 5$	$N = 9$

(d) feature 'cell' and category 'spam'

	$v_c = v_{spam} = 1$	$v_c = v_{spam} = 0$	
$v_f = v_{ski} = 1$	$N_{F,C_k} = 0$	$N_{F,\overline{C_k}} = 3$	$N_F = 3$
$v_f = v_{ski} = 0$	$N_{\overline{F},C_k} = 4$	$N_{\overline{F},\overline{C_k}} = 2$	$N_{\overline{F}} = 6$
	$N_{C_k} = 4$	$N_{\overline{C_k}} = 5$	$N = 9$

(e) feature 'ski' and category 'spam'

Table 3.3: Contingency tables for the e-mail example (the spam category). The tables contain document counts, see Table 3.1 for explanation.

	$v_c = v_{business} = 1$	$v_c = v_{business} = 0$	
$v_f = v_{viagra} = 1$	$N_{F,C_k} = 0$	$N_{F,\overline{C_k}} = 5$	$N_F = 5$
$v_f = v_{viagra} = 0$	$N_{\overline{F},C_k} = 3$	$N_{\overline{F},\overline{C_k}} = 1$	$N_{\overline{F}} = 4$
	$N_{C_k} = 3$	$N_{\overline{C_k}} = 6$	$N = 9$

(a) feature 'viagra' and category 'business'

	$v_c = v_{business} = 1$	$v_c = v_{business} = 0$	
$v_f = v_{save} = 1$	$N_{F,C_k} = 2$	$N_{F,\overline{C_k}} = 1$	$N_F = 3$
$v_f = v_{save} = 0$	$N_{\overline{F},C_k} = 1$	$N_{\overline{F},\overline{C_k}} = 5$	$N_{\overline{F}} = 6$
	$N_{C_k} = 3$	$N_{\overline{C_k}} = 6$	$N = 9$

(b) feature 'save' and category 'business'

	$v_c = v_{business} = 1$	$v_c = v_{business} = 0$	
$v_f = v_{erection} = 1$	$N_{F,C_k} = 0$	$N_{F,\overline{C_k}} = 3$	$N_F = 3$
$v_f = v_{erection} = 0$	$N_{\overline{F},C_k} = 3$	$N_{\overline{F},\overline{C_k}} = 3$	$N_{\overline{F}} = 6$
	$N_{C_k} = 3$	$N_{\overline{C_k}} = 6$	$N = 9$

(c) feature 'erection' and category 'business'

	$v_c = v_{business} = 1$	$v_c = v_{business} = 0$	
$v_f = v_{cell} = 1$	$N_{F,C_k} = 2$	$N_{F,\overline{C_k}} = 0$	$N_F = 2$
$v_f = v_{cell} = 0$	$N_{\overline{F},C_k} = 1$	$N_{\overline{F},\overline{C_k}} = 6$	$N_{\overline{F}} = 7$
	$N_{C_k} = 3$	$N_{\overline{C_k}} = 6$	$N = 9$

(d) feature 'cell' and category 'business'

	$v_c = v_{business} = 1$	$v_c = v_{business} = 0$	
$v_f = v_{ski} = 1$	$N_{F,C_k} = 1$	$N_{F,\overline{C_k}} = 2$	$N_F = 3$
$v_f = v_{ski} = 0$	$N_{\overline{F},C_k} = 2$	$N_{\overline{F},\overline{C_k}} = 4$	$N_{\overline{F}} = 6$
	$N_{C_k} = 3$	$N_{\overline{C_k}} = 6$	$N = 9$

(e) feature 'ski' and category 'business'

Table 3.4: Contingency tables for the e-mail example (the business category). The tables contain document counts, see Table 3.1 for explanation.

	$v_c = v_{private} = 1$	$v_c = v_{private} = 0$	
$v_f = v_{viagra} = 1$	$N_{F,C_k} = 1$	$N_{F,\overline{C_k}} = 4$	$N_F = 5$
$v_f = v_{viagra} = 0$	$N_{\overline{F},C_k} = 1$	$N_{\overline{F},\overline{C_k}} = 3$	$N_{\overline{F}} = 4$
	$N_{C_k} = 2$	$N_{\overline{C_k}} = 7$	$N = 9$

(a) feature 'viagra' and category 'private'

	$v_c = v_{private} = 1$	$v_c = v_{private} = 0$	
$v_f = v_{save} = 1$	$N_{F,C_k} = 0$	$N_{F,\overline{C_k}} = 3$	$N_F = 3$
$v_f = v_{save} = 0$	$N_{\overline{F},C_k} = 2$	$N_{\overline{F},\overline{C_k}} = 4$	$N_{\overline{F}} = 6$
	$N_{C_k} = 2$	$N_{\overline{C_k}} = 7$	$N = 9$

(b) feature 'save' and category 'private'

	$v_c = v_{private} = 1$	$v_c = v_{private} = 0$	
$v_f = v_{erection} = 1$	$N_{F,C_k} = 0$	$N_{F,\overline{C_k}} = 3$	$N_F = 3$
$v_f = v_{erection} = 0$	$N_{\overline{F},C_k} = 2$	$N_{\overline{F},\overline{C_k}} = 4$	$N_{\overline{F}} = 6$
	$N_{C_k} = 2$	$N_{\overline{C_k}} = 7$	$N = 9$

(c) feature 'erection' and category 'private'

	$v_c = v_{private} = 1$	$v_c = v_{private} = 0$	
$v_f = v_{cell} = 1$	$N_{F,C_k} = 0$	$N_{F,\overline{C_k}} = 2$	$N_F = 2$
$v_f = v_{cell} = 0$	$N_{\overline{F},C_k} = 2$	$N_{\overline{F},\overline{C_k}} = 5$	$N_{\overline{F}} = 7$
	$N_{C_k} = 2$	$N_{\overline{C_k}} = 7$	$N = 9$

(d) feature 'cell' and category 'private'

	$v_c = v_{private} = 1$	$v_c = v_{private} = 0$	
$v_f = v_{ski} = 1$	$N_{F,C_k} = 2$	$N_{F,\overline{C_k}} = 1$	$N_F = 3$
$v_f = v_{ski} = 0$	$N_{\overline{F},C_k} = 0$	$N_{\overline{F},\overline{C_k}} = 6$	$N_{\overline{F}} = 6$
	$N_{C_k} = 2$	$N_{\overline{C_k}} = 7$	$N = 9$

(e) feature 'ski' and category 'private'

Table 3.5: Contingency tables for the e-mail example (the private category). The tables contain document counts, see Table 3.1 for explanation.

3.6 Unsupervised Feature Selection

Unsupervised feature selection methods are methods that do not use the class information in the training data when selecting features for the classifier. This does not mean that a training set is not necessary, but it means that the training data does not need to be manually pre-classified. All that is needed is to index a fixed set of documents from the collection the classifier is to be used on. Hence, these methods are handy if there is no pre-classified training data available, and if there is no time to create such data. Pre-classified documents are of course needed for evaluation of the classifier's performance, however.

This chapter will present earlier, related work in unsupervised feature selection methods, as well as a few methods not used for feature selection before. We will run them through our e-mail sorting example where applicable.

We will not use any denomination on the feature values created by the feature selection metrics. This is because we are only interested in the order the features are ranked. The actual values are not used after the ranking is done.

3.6.1 Random Feature Selection

The simplest way of reducing the number of features, is to randomly select as many features as needed. Since random selection naturally does no attempt of finding good features, its performance is not expected to be good either. We employ it as a reference figure, as is common practice in feature selection comparison papers.

Note that since the features are selected randomly, the performance should not be expected to follow an evenly rising curve as the number of features selected increase. The extra features added can easily be more noisy than the previously selected ones. Also, each generation of a randomly ranked feature list will be different, and hence it would be impossible to recreate our results.

3.6.2 Collection Frequency (CF)

The collection frequency of a feature is the total number of instances of the feature in the collection. It does not look at which documents or categories the feature occurs in, it is simply a count. If a feature occurs twice in a document, it gets counted twice, and so on.

As most feature selection methods are concerned with document counts, so is our notation presented in Section 3.4. Hence, we cannot use that notation to present this method. However, our example presented in Table 3.2 has some documents with a feature repeated up to three times. Table 3.6 ranks the features in our example by their collection frequency value.

3.6.3 Collection Frequency Inverse Document Frequency (CFIDF)

The CFIDF is computed by weighting the collection frequency values by the inverse document frequency for a feature:

$$CFIDF(F) = CF(F) \log_2 \frac{N}{DF(F)} \tag{3.1}$$

Feature	Collection Frequency value
viagra	6.0
cell	4.0
save	3.0
erection	3.0
ski	3.0

Table 3.6: Features ranked by Collection Frequency value.

where the collection frequency is counted as explained in Section 3.6.2, and N is the total number of documents in the training data. The inverse document frequency is discussed briefly in Section 2.3.2.

The motivation behind this metric is that the combination of the local document frequency and total number of occurrences for a feature could provide a better ranking than the collection frequency alone.

This method makes use of a logarithm statement. Which logarithm base to use (e.g. $\log_2$ or ln) is not important, as the ranking stays the same. We use $\log_2$ here, as this is common in tf-idf computation (tf-idf is discussed in Section 2.3.2), and we reused tf-idf code in this method.

Example

We compute the CFIDF feature values for the e-mail example:

$$CFIDF(viagra) = 6 \times \log_2 \frac{9}{5} = \underline{5.0880}$$

$$CFIDF(save) = 3 \times \log_2 \frac{9}{3} = \underline{4.7549}$$

$$CFIDF(erection) = 3 \times \log_2 \frac{9}{3} = \underline{4.7549}$$

$$CFIDF(cell) = 4 \times \log_2 \frac{9}{2} = \underline{8.6797}$$

$$CFIDF(ski) = 3 \times \log_2 \frac{9}{3} = \underline{4.7549}$$

Table 3.7 ranks the e-mail example features according to their CFIDF value. Note that CFIDF values are naturally global, so there is no need to aggregate them in any way.

Feature	CFIDF value
cell	8.6797
viagra	5.0880
save	4.7549
erection	4.7549
ski	4.7549

Table 3.7: Features ranked by CFIDF value.

3.6.4 Document Frequency (DF) Thresholding

One of the simplest methods of vocabulary reduction, and hence vector dimensionality reduction, is the Document Frequency Thresholding, presented in Equation (3.2).

$$DF(F) = N_F \tag{3.2}$$

The number of documents containing a feature in the training set is counted. This is done for every feature in the training set, before removing all features with a document frequency less than some specified threshold and features with a frequency higher than some other threshold. Alternatively, the document frequency can be used as any other feature selection method where it creates a ranked list, and returns the i highest ranked features.

Example Setup

The document frequency values for our e-mail example can be read directly from Tables 3.3, 3.4 and 3.5 (or counted from Table 3.2).

Table 3.8 ranks the e-mail example features according to their document frequency value. Note that document frequency values are naturally global, so there is no need to aggregate them in any way.

Feature	Document Frequency value
viagra	5.0
save	3.0
erection	3.0
ski	3.0
cell	2.0

Table 3.8: Features ranked by Document Frequency value.

3.6.5 Term Frequency Document Frequency (TFDF)

In [XWLJ08], a method based on the term frequency combined with the document frequency threshold (Section 3.6.4) is presented. They call it Term Frequency Document Frequency, and prove it better than DF thresholding.

Equations

$$TFDF(F) = (n_1 \times n_2 + c(n_1 \times n_3 + n_2 \times n_3)) \tag{3.3}$$

where c is a constant $c \geq 1$, n_1 is the number of documents without the feature, n_2 is the number of documents where the feature occurs exactly once, n_3 is the number of documents where the feature occurs twice or more.

The authors of [XWLJ08] use $c = 10$ in their experiments, and we follow this decision in our experiments. It should be noted however, that the constant can highly affect the results. Hence, in an operational setting, performance should be measured for several levels of the constant, with the actual text collection and

classification learner at hand, as to achieve the most from this feature selection method.

Example Setup

The n-values can be extracted from Table 3.2. We list them here for clarity:

$$\begin{aligned}
\text{viagra: } & n_1 = 4, \;\; n_2 = 4, \;\; n_3 = 1 \\
\text{save: } & n_1 = 6, \;\; n_2 = 3, \;\; n_3 = 0 \\
\text{erection: } & n_1 = 6, \;\; n_2 = 3, \;\; n_3 = 0 \\
\text{cell: } & n_1 = 7, \;\; n_2 = 1, \;\; n_3 = 1 \\
\text{ski: } & n_1 = 6, \;\; n_2 = 3, \;\; n_3 = 0
\end{aligned}$$

We set $c = 1$ and calculate the TFDF values for the example:

$$\begin{aligned}
TFDF(viagra) &= (4 \times 4 + 1(4 \times 1 + 4 \times 1)) \\
&= 16 + 8 = \underline{24} \\
TFDF(save) &= (6 \times 3 + 1(6 \times 0 + 3 \times 0)) \\
&= 18 + 0 = \underline{18} \\
TFDF(erection) &= (6 \times 3 + 1(6 \times 0 + 3 \times 0)) \\
&= 18 + 0 = \underline{18} \\
TFDF(cell) &= (7 \times 1 + 1(7 \times 1 + 1 \times 1)) \\
&= 7 + 8 = \underline{15} \\
TFDF(ski) &= (6 \times 3 + 1(6 \times 0 + 3 \times 0)) \\
&= 18 + 0 = \underline{18}
\end{aligned}$$

We then set $c = 10$ and calculate the TFDF values for the example:

$$\begin{aligned}
TFDF(viagra) &= (4 \times 4 + 10(4 \times 1 + 4 \times 1)) \\
&= 16 + 80 = \underline{96} \\
TFDF(save) &= (6 \times 3 + 10(6 \times 0 + 3 \times 0)) \\
&= 18 + 0 = \underline{18} \\
TFDF(erection) &= (6 \times 3 + 10(6 \times 0 + 3 \times 0)) \\
&= 18 + 0 = \underline{18} \\
TFDF(cell) &= (7 \times 1 + 10(7 \times 1 + 1 \times 1)) \\
&= 7 + 80 = \underline{87} \\
TFDF(ski) &= (6 \times 3 + 10(6 \times 0 + 3 \times 0)) \\
&= 18 + 0 = \underline{18}
\end{aligned}$$

We can see clearly with the example that the features that occur more than once are favored with this feature selection scheme, and more so as the value of c increases.

Table 3.9 ranks the e-mail example features according to their global word frequency value when the constant $c = 1$:

Table 3.10 ranks the e-mail example features according to their global word frequency value when the constant $c = 10$:

Feature	(global) TFDF value ($c = 1$)
viagra	24.0
erection	18.0
save	18.0
ski	18.0
cell	15.0

Table 3.9: Features ranked by Word Frequency Document Frequency (TFDF) value when the constant $c = 1$.

Feature	(global) TFDF value ($c = 10$)
viagra	96.0
cell	87.0
erection	18.0
save	18.0
ski	18.0

Table 3.10: Features ranked by Word Frequency Document Frequency (TFDF) value when the constant $c = 10$.

Notice how the feature 'cell' now has been boosted up from last to second place in the ranking, because of its triple occurrence in document #7 in the e-mail example.

3.6.6 Weirdness Factor

The weirdness factor was presented in [AGT99] for use in TREC8. It calculates the relative document frequency for the features, and compares them to their relative document frequencies in a reference corpus (with no special domain or topic). This way, features that are used frequently in normal texts receive low weirdness, while more specialized terms receive higher rating.

As far as we know, the weirdness factor has not been used for feature selection for text categorization before, and we are want to see if it can compete with the well-known methods.

Equation

The Weirdness factor of a feature is computed by Equation (3.7).

$$Weirdness(F) = \frac{\frac{w_s}{t_s}}{\frac{w_g}{t_g}} \tag{3.4}$$

where w_s is is the frequency of the feature in the corpus to be classified, w_g is the frequency of the feature in the general reference corpus, t_s is the total count of the feature in the corpus to be classified, and t_g is the total count of that feature in the general reference corpus.

We have used the number of documents a feature occurs in (DF) instead of the total number of feature occurrences (CF) for the t_s and t_g values.

We give features that do not occur at all in the reference corpus an average frequency value from that corpus. This makes miss-spelled words, e-mail addresses etc. not stand out with very high weirdness factors, but rather puts them in the middle of the list, based on how often they occur. As such, they will not be selected when using aggressive feature selection, but will be ranked above commonly used words.

Example Setup

Since we use the BNC reference corpus in our experiments, we will use it in the e-mail filter example as well. Note that the weirdness factor is global by nature, i.e. it produces one list per corpus, and not one per category.

$$Weirdness(viagra) = \frac{\frac{5.0}{9.0}}{\frac{25.5692}{4049.0}} = \underline{87.9749}$$

$$Weirdness(ski) = \frac{\frac{3.0}{9.0}}{\frac{397}{4049.0}} = \underline{3.3997}$$

$$Weirdness(save) = \frac{\frac{3.0}{9.0}}{\frac{2699.0}{4049.0}} = \underline{0.5001}$$

$$Weirdness(erection) = \frac{\frac{3.0}{9.0}}{\frac{25.5692}{4049.0}} = \underline{52.7849}$$

$$Weirdness(cell) = \frac{\frac{2.0}{9.0}}{\frac{962.0}{4049.0}} = \underline{0.9353}$$

Table 3.11 ranks the e-mail example features according to their weirdness factor compared to the BNC reference corpus:

Feature	**Weirdness factor**
viagra	87.9749
erection	52.7849
ski	3.3997
cell	0.9353
save	0.5001

Table 3.11: Features ranked by weirdness factor compared to the BNC reference corpus.

3.7 Supervised Feature Selection

Supervised feature selection methods make use of the category information in the training data. Hence, for supervised methods to be usable, a pre-classified set of documents must be available. Thus it is natural to expect better performance from these methods than from the unsupervised methods presented in Section 3.6.

This chapter will present earlier, related work in supervised feature selection methods. We will run them through our e-mail sorting example where applicable.

Some of the methods presented here use logarithm statements in their equations. We may have used other logarithm bases than other researchers. This is not important however, as we only are concerned about the ranking and not the actual feature values computed. Although the feature values change when the logarithm base changes, the ranking stays the same. For the same reason, we will not use any denomination on the feature values computed by the feature selection metrics. Some methods are sometimes denominated in the literature, for instance the IG metric may be denominated bits if $\log_2$ is used. We choose to skip all denomination, as it can be more confusing than clarifying.

3.7.1 Word Frequency (WF)

The simple Word Frequency for a (feature, category) pair is defined by [EM07] as the number of documents in category C_k containing feature F, as shown in Equation (3.5). Hence it looks only for positive evidence of category membership.

$$WF(F, C_k) = N_{F,C_k} \tag{3.5}$$

As done in [EM07], we aggregate these values to find the global Word Frequency value for each feature F by weighting the value of each (feature, category) pair by the category dominance and then summarize the weighted values:

$$WF(F) = \sum_{k=1}^{|C|} \frac{N_{C_k}}{N} N_{F,C_k} \tag{3.6}$$

Example Setup

The category-local values used in Equation (3.5) can be read directly from Tables 3.3, 3.4 and 3.5. Using Equation (3.6), we calculate the global word frequencies for each feature:

$$
\begin{aligned}
WF(viagra) &= \sum_{k=1}^{C} \frac{N_{C_k}}{N} N_{F,C_k} \\
&= \frac{N_{C_k=sp.}}{N} N_{F,C_k=sp.} \\
&\quad + \frac{N_{C_k=bus.}}{N} N_{F,C_k=bus.} \\
&\quad + \frac{N_{C_k=priv.}}{N} N_{F,C_k=priv.} \\
&= \frac{4}{9} \times 4 + \frac{3}{9} \times 0 + \frac{2}{9} \times 1 = \underline{2.0} \\
WF(save) &= \frac{4}{9} \times 1 + \frac{3}{9} \times 2 + \frac{2}{9} \times 0 = \frac{10}{9} = \underline{1.1111} \\
WF(erection) &= \frac{4}{9} \times 3 + \frac{3}{9} \times 0 + \frac{2}{9} \times 0 = \frac{4}{3} = \underline{1.3333} \\
WF(cell) &= \frac{4}{9} \times 0 + \frac{3}{9} \times 2 + \frac{2}{9} \times 0 = \frac{2}{3} = \underline{0.6667} \\
WF(ski) &= \frac{4}{9} \times 0 + \frac{3}{9} \times 1 + \frac{2}{9} \times 2 = \frac{7}{9} = \underline{0.7778}
\end{aligned}
$$

Table 3.12 ranks the e-mail example features according to their global word frequency (WF) value.

Feature	**Global Word Frequency value**
viagra	2.0000
erection	1.3333
save	1.1111
ski	0.7778
cell	0.6667

Table 3.12: Features ranked by Word Frequency value.

3.7.2 Information Gain (IG)

The basic idea behind IG is to find out how well each single feature separates the given data set. Information entropy is used to measure the uncertainty of the feature (e.g. term) and the dataset (e.g. a corpus of documents).

Equation

The Information Gain of a feature is computed by Equation (3.7).

$$\begin{aligned} IG(Feature) = &- \sum_{k=1}^{C} \frac{N_{C_k}}{N} \ln \frac{N_{C_k}}{N} \\ &+ \frac{N_F}{N} \sum_{k=1}^{C} \frac{N_{F,C_k}}{N_F} \ln \frac{N_{F,C_k}}{N_F} \\ &+ \frac{N_{\overline{F}}}{N} \sum_{k=1}^{C} \frac{N_{\overline{F},C_k}}{N_{\overline{F}}} \ln \frac{N_{\overline{F},C_k}}{N_{\overline{F}}} \end{aligned} \tag{3.7}$$

Equation (3.7) takes the overall entropy for the training set (the first line of the equation) minus the entropy for the feature (the last two lines of the equation). In Equation (3.7), we calculate the expected reduction in entropy if we categorize the corpus according to that feature. After computing IG values for all features, we can use the features with the highest IG score as features in any text categorization classifier.

Notice that the overall entropy for the training set naturally is the same for all features. Hence Equation (3.7) could easily have been split up into the calculation of the overall entropy $H(Set)$ and the feature entropy values $H(Feature)$, and then for each feature the calculation of the Information Gain value $IG(Feature) = H(Set) - H(Feature)$.

Background

Information Gain was originally developed by J. Ross Quinlan [Qui86] for inducing decision trees (the ID3 algorithm – Iterative Dichotomizer[2] 3). For our task of feature selection, we do not need the decision trees, but rather the Information Gain values from equation (3.7) for each feature[3].

The entropy values are normally denominated 'bits' if logarithm base 2 is used, as for how many bits are necessary for encoding the information piece on average. We will skip the denomination here, however, as we only use the computed values for *ranking* a list of terms.

Example Setup

We estimate the probabilities by counting the manually categorized documents from the training set. The N_{fc} notation is explained in Section 3.4, and their values are collected from Tables 3.3, 3.4 and 3.5.

Using equation (3.7), we can now calculate the Information Gain value for the feature 'viagra'.

[2] Dichotomy is a division into exactly two mutually exclusive categories.

[3] In fact, we do not even need the information gain value. A reversed list of information entropy values would work just as well for our feature ranking purpose. However, since most literature on feature selection uses the full IG value computed, so will we.

$$
\begin{aligned}
IG(Viagra) = &- \sum_{k=1}^{C} \frac{N_{C_k}}{N} \ln \frac{N_{C_k}}{N} \\
&+ \frac{N_F}{N} \sum_{k=1}^{C} \frac{N_{F,C_k}}{N_F} \ln \frac{N_{F,C_k}}{N_F} \\
&+ \frac{N_{\overline{F}}}{N} \sum_{k=1}^{C} \frac{N_{\overline{F},C_k}}{N_{\overline{F}}} \ln \frac{N_{\overline{F},C_k}}{N_{\overline{F}}} \\
= &- \left(\frac{4}{9} \times \ln \frac{4}{9}\right) - \left(\frac{3}{9} \times \ln \frac{3}{9}\right) - \left(\frac{2}{9} \times \ln \frac{2}{9}\right) \\
&+ \frac{5}{9} \times \left(\frac{4}{5} \ln \frac{4}{5} + \frac{0}{5} \ln \frac{0}{5} + \frac{1}{5} \ln \frac{1}{5}\right) \\
&+ \frac{4}{9} \times \left(\frac{0}{4} \ln \frac{0}{4} + \frac{3}{4} \ln \frac{3}{4} + \frac{1}{4} \ln \frac{1}{4}\right) \\
= &- (-0.3604) - (-0.3662) - (-0.3342) \\
&+ \frac{5}{9} \times (-0.1785 + 0 - 0.3219) \\
&+ \frac{4}{9} \times (0 - 0.2158 - 0.3466) \\
= &1.0609 - 0.2780 - 0.2499 = \underline{0.5330}
\end{aligned}
$$

where the N-values are collected from Tables 3.3, 3.4 and 3.5.

The Information Gain for *save* now calculates as follows, using equation (3.7):

$$
\begin{aligned}
IG(save) = 1.0609 + &\frac{3}{9}\left(\frac{1}{3} \ln \frac{1}{3} + \frac{2}{3} \ln \frac{2}{3} + \frac{0}{3} \ln \frac{0}{3}\right) \\
+ &\frac{6}{9}\left(\frac{3}{6} \ln \frac{3}{6} + \frac{1}{6} \ln \frac{1}{6} + \frac{2}{6} \ln \frac{2}{6}\right) \\
= 1.0609 - &0.2122 - 0.6742 = \underline{0.1744}
\end{aligned}
$$

Notice that we used the value for the overall entropy directly from the calculation of $IG(viagra)$ above.

Similarly, the Information Gain value for the *erection* feature now calculates as follows, again using equation (3.7).

$$
\begin{aligned}
IG(erection) = 1.0609 + &\frac{3}{9}\left(\frac{3}{3} \ln \frac{3}{3} + \frac{0}{3} \ln \frac{0}{3} + \frac{0}{3} \ln \frac{0}{3}\right) \\
+ &\frac{6}{9}\left(\frac{1}{6} \ln \frac{1}{6} + \frac{3}{6} \ln \frac{3}{6} + \frac{2}{6} \ln \frac{2}{6}\right) \\
= 1.0609 + &0 - 0.6743 = \underline{0.3866}
\end{aligned}
$$

and we can see that we have gained all the information available with this feature alone. That is, the 'erection' feature is alone enough to completely separate the corpus.

We skip the computation details for the *cell* and *ski* features. The features are rated according to their IG value in Table 3.13. Note that Information Gain is global by nature, so there is no need to aggregate these values.

Feature	Information Gain value
viagra	0.5330
ski	0.4244
erection	0.3866
cell	0.3176
save	0.1744

Table 3.13: Features ranked by information gain (IG) value.

3.7.3 Mutual Information (MI)

Mutual Information can be proven equal to Information Gain for binary problems. For mutli-class problems (with global feature lists) like we present in this report however, the two are not equal (although rather similar). Thus we present Mutual Information with it's own equation as a separate feature selection algorithm here.

Equations

We compute the Mutual Information of a term and category pair as shown in Equation (3.8):

$$MI(F, C_k) = \sum_{v_f \epsilon \{1,0\}} \sum_{v_{C_k} \epsilon \{1,0\}} P(F = v_f, C_k = v_{C_k}) \ln \frac{P(F = v_f, C_k = v_{C_k})}{P(F = v_f) P(C_k = v_{C_k})} \tag{3.8}$$

where F is the discrete random variable 'feature' that takes the value $v_F = \{1, 0\}$ (feature F occurs in document or not), C_k is the discrete random variable 'category' that takes the values $v_{C_k} = \{1, 0\}$ (document belongs to category C_k or not).

The probabilities can be estimated by using the various document counts from the training set. Using our own notation from Section 3.4 we rewrite Equation (3.8) into Equation (3.9):

$$\begin{aligned} MI(F, C_k) = & \frac{N_{F,C_k}}{N} \ln \frac{N N_{F,C_k}}{N_F N_{C_k}} + \frac{N_{F,\overline{C_k}}}{N} \ln \frac{N N_{F,\overline{C_k}}}{N_F N_{\overline{C_k}}} \\ & + \frac{N_{\overline{F},C_k}}{N} \ln \frac{N N_{\overline{F},C_k}}{N_{\overline{F}} N_{C_k}} + \frac{N_{\overline{F},\overline{C_k}}}{N} \ln \frac{N N_{\overline{F},\overline{C_k}}}{N_{\overline{F}} N_{\overline{C_k}}} \end{aligned} \tag{3.9}$$

Then the values can be weighted and summarized to create a global ranked list of features:

$$MI(F) = \sum_{k=1}^{|C|} \frac{N_{C_k}}{N} MI(F, C_k) \tag{3.10}$$

Naming Confusion

As the IG and MI equations are equal for binary problems, there exists some naming confusion in the literature. Equation (3.8) is used with the name Mutual Information (like we do) or *expected* Mutual Information in [MRS08, MK07,

DC00] and [vR79, Chapter 3]. It is used with the name Information Gain in [BMGMF08, FS07, DDH+07, Seb02].

Also, Equation (3.7) as we presented under the Information Gain name in Section 3.7.2, is named Mutual Information in [Joa97]. It is used with the Information Gain name in [SH08, MRS08, EM07, MK07, YP97].

Moreover, Equation (3.11) as we present as *Pointwise* Mutual Information in Section 3.7.4, is used with the name Mutual Information in [SH08, Seb02, YP97], while we follow [MRS08].

Example

We calculate the mutual information values for the e-mail example in Table 3.2 using equation (3.9). We show here the calculations for the 'spam' category, using the values from Table 3.3. Natural log (ln) is used here, i.e. the base is e. However, we note that any logarithm can be used, as we are only interested in the ranking order, and not the actual feature values.

$$\begin{aligned}
MI(viagra; spam) =& \frac{4}{9}\ln\frac{9\times 4}{5\times 4} + \frac{1}{9}\ln\frac{9\times 1}{5\times 5} \\
&+\frac{0}{9}\ln\frac{9\times 0}{4\times 4} + \frac{4}{9}\ln\frac{9\times 4}{4\times 5} = \underline{0.4090} \\
MI(save; spam) =& \frac{1}{9}\ln\frac{9\times 1}{3\times 4} + \frac{2}{9}\ln\frac{9\times 2}{3\times 5} \\
&+\frac{3}{9}\ln\frac{9\times 3}{6\times 4} + \frac{3}{9}\ln\frac{9\times 3}{6\times 5} = \underline{0.0127} \\
MI(erection; spam) =& \frac{3}{9}\ln\frac{9\times 3}{3\times 4} + \frac{0}{9}\ln\frac{9\times 0}{3\times 5} \\
&+\frac{1}{9}\ln\frac{9\times 1}{6\times 4} + \frac{5}{9}\ln\frac{9\times 5}{6\times 5} = \underline{0.3866} \\
MI(cell; spam) =& \frac{0}{9}\ln\frac{9\times 0}{2\times 4} + \frac{2}{9}\ln\frac{9\times 2}{2\times 5} \\
&+\frac{4}{9}\ln\frac{9\times 4}{7\times 4} + \frac{3}{9}\ln\frac{9\times 3}{7\times 5} = \underline{0.1558} \\
MI(ski; spam) =& \frac{0}{9}\ln\frac{9\times 0}{3\times 4} + \frac{3}{9}\ln\frac{9\times 3}{3\times 5} \\
&+\frac{4}{9}\ln\frac{9\times 4}{6\times 4} + \frac{2}{9}\ln\frac{9\times 2}{6\times 5} = \underline{0.2626}
\end{aligned}$$

Similarly, we calculated the mutual information values for each feature combined with the 'business' and 'private' categories, using equation (3.9) and values from Tables 3.4 and 3.5, respectively. The results are presented in Table 3.14.

3.7.4 Pointwise Mutual Information (PMI)

Pointwise Mutual Information has been proven as a weak feature selection method because of its bias to favoring rare features [YP97].

Term	Spam	Business	Private	Sum	wAvg
viagra	0.4090	0.3866	0.0018	0.7974	0.3110
erection	0.3866	0.1744	0.1054	0.6664	0.2533
cell	0.1558	0.3175	0.0644	0.5377	0.1894
ski	0.2626	0.0000	0.3175	0.5801	0.1873
save	0.0127	0.1240	0.1054	0.2421	0.0704

Table 3.14: E-mail example: Features ranked by Mutual Information (MI) value.

Equations

Yang and Pedersen [YP97] present the equation as in Equation (3.11):

$$I(t,c) = \log \frac{Pr(t \wedge c)}{Pr(t) \times Pr(c)} \tag{3.11}$$

where Pr are probabilities, t is for term (feature), and c is for category. If we write this with our own syntax, we get equation (3.12).

$$PMI(F, C_k) = \log \frac{N_{F,C_k} \times N}{N_F \times N_{C_k}} \tag{3.12}$$

In [YP97] this method was called Mutual Information. We follow [MRS08, footnote on page 272] and name it *Pointwise* Mutual Information. The name Mutual Information is later also used on this method in [SH08] and [Seb02]. We have not tested this metric, as it is known for its bad results. We merely present it for clarity regarding the Mutual Information naming.

3.7.5 Odds Ratio (OR)

Odds Ratio [Sha95, Mla98, CHTQ09] compares the odds of a feature occurring in one category with the odds for it occurring in another category. It gives a positive score to features that occur more often in one category than in the other, and a negative score if it occurs more in the other. A score of zero means the the odds for a feature to occur in one category is exactly the same as the odds for it to occur in the other, since $\ln(1) = 0$.

Equation

The original Odds Ratio algorithm for binary categorization:

$$OR(F, C_k) = \ln \frac{P(F|C_k)(1 - P(F|\overline{C_k}))}{P(F|\overline{C_k})(1 - P(F|C_k))} = \ln \frac{\left(\frac{N_{F,C_k}}{N_{C_k}}\right)\left(1 - \frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)}{\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)\left(1 - \frac{N_{F,C_k}}{N_{C_k}}\right)} \tag{3.13}$$

where F is a feature, C_k is the category of concern, $P(F|C_k)$ is the probability for the feature F to occur in category C_k, and $P(F|\overline{C_k})$ is the probability for the feature F to occur in category $\overline{C_k}$.

To estimate the probabilities, we use the number of training documents in category C_k containing the feature F divided by the total number of training documents in category C_k, and similarly for category $\overline{C_k}$.

$$P(F|C_k) = \frac{N_{F,C_k}}{N_{C_k}}$$

An alert reader will notice that there might occur divide-by-zero and $\ln(0)$ problems when using this estimation technique with equation (3.13). We follow [Sha95] and treat singularities as special cases: When $P(F|C_k) = 0$ because none of the training documents in category C_k contain the feature F, we substitute $P(F|C_k)$ with $\frac{1}{N^2}$. Also, when $P(F|C_k) = 1$ because *all* the training documents in category C_k contains the feature F, we substitute $P(F|C_k)$ with $1 - \frac{1}{N^2}$, where N is the number of documents in the whole corpus/collection.

Thus the equation for estimating the probabilities including the special cases becomes this:

$$P(F|C_k) = \begin{cases} \frac{N_{F,C_k}}{N_{C_k}} & \\ \frac{1}{N^2} & \text{if } N_{F,C_k} = 0 \\ 1 - \frac{1}{N^2} & \text{if } N_{F,C_k} = N_{C_k} \end{cases}$$

Taking the square of the corpus size into consideration ensures that low probabilities are well estimated in small corpora.

Background and Explanation

The odds of a feature occurring in exactly one of two categories $\text{Odds}(F|C_k)$ are the quantity $\frac{P(F|C_k)}{(1-P(F|C_k))}$, where $P(F|C_k)$ is the probability for the feature F to occur in category C_k. Hence the odds are the *relative* probability, which is important since we are to find the features that carry much information. A feature that occurs often in one category is not a good feature if it occurs just as frequently in the other.

The odds *ratio* is the odds for one event divided by the odds for the opposite event(s). In our case, the odds ratio is the odds for a feature occurring in one category divided by the odds for it to occur in the other category, since we are now talking about binary classification (e.g. spam or not spam).

The odds ratio equations we use are surrounded by a natural logarithm statement. This is not necessary to calculate the odds ratio itself, but it has several desired properties. First, it separates good odds ratios from poor odds ratios with positive and negative values (while the basic odds ratio equation without logarithm would only calculate positive values). Second, the logarithm reduces positive skew by compressing the tail (high values) and expanding the head (low values).

Thus we find the Odds Ratio formula the following way. Note that the

notation is explained in Section 3.4.

$$OR(f, C_k) = \ln \frac{\text{Odds}(F|C_k)}{\text{Odds}(F|\overline{C_k})} = \ln \frac{\left(\frac{P(F|C_k)}{(1-P(F|C_k))}\right)}{\left(\frac{P(F|\overline{C_k})}{(1-P(F|\overline{C_k}))}\right)} = \ln \frac{\left(\frac{\frac{N_{F,C_k}}{N_{C_k}}}{1-\frac{N_{F,C_k}}{N_{C_k}}}\right)}{\left(\frac{\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}}{1-\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}}\right)}$$

$$= \ln \frac{P(F|C_k)(1-P(F|\overline{C_k}))}{P(F|\overline{C_k})(1-P(F|C_k))} = \ln \frac{\left(\frac{N_{F,C_k}}{N_{C_k}}\right)\left(1-\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)}{\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)\left(1-\frac{N_{F,C_k}}{N_{C_k}}\right)}$$

The higher values, the better the term should be for deciding if documents belong in the considered category. Hence for feature selection for text categorization, we rank the terms by log Odds Ratio values, and use the k top ranked terms in some machine learning algorithm.

Example

We run the Odds Ratio algorithm on the e-mail example from Section 3.5. We start by calculating the Odds Ratio for the feature 'viagra' using Equation (3.13) and values from Table 3.3:

$$OR(viagra, spam) = \ln \frac{P(viagra|spam)(1-P(viagra|\overline{spam}))}{P(viagra|\overline{spam})(1-P(viagra|spam))}$$

$$= \ln \frac{\left(\frac{N_{F,C_k}}{N_{C_k}}\right)\left(1-\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)}{\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)\left(1-\frac{N_{F,C_k}}{N_{C_k}}\right)} = \ln \frac{(\frac{4}{4})(1-\frac{1}{5})}{(\frac{1}{5})(1-\frac{4}{4})}$$

but when considering singularities:

$$= \ln \frac{(1-\frac{1}{N^2})(1-\frac{1}{5})}{(\frac{1}{5})(1-(1-\frac{1}{N^2}))} = \ln \frac{(1-\frac{1}{81})(1-\frac{1}{5})}{(\frac{1}{5})(1-(1-\frac{1}{81}))}$$

$$= \ln \frac{(\frac{80}{81})(\frac{4}{5})}{(\frac{1}{5})(\frac{1}{81})} = \ln \frac{(\frac{64}{81})}{(\frac{1}{405})} = \ln(320) = \underline{5.7683}$$

We calculate the Odds Ratio for the other features in the 'spam' category in the same way, using equation (3.13) and values from Table 3.3:

$$OR(save, spam) = \ln \frac{(\frac{1}{4})(1-\frac{2}{5})}{(\frac{2}{5})(1-\frac{1}{4})} = \ln\left(\frac{1}{2}\right) = \underline{-0.6931}$$

$$OR(erection, spam) = \ln \frac{(\frac{3}{4})(1-\frac{1}{81})}{(\frac{1}{81})(1-\frac{3}{4})} = \ln(240) = \underline{5.4806}$$

$$OR(cell, spam) = \ln \frac{(\frac{1}{81})(\frac{3}{5})}{(\frac{2}{5})(1-\frac{1}{81})} = \ln\left(\frac{3}{160}\right) = \underline{-3.9766}$$

$$OR(ski, spam) = \ln \frac{(\frac{1}{81})(1-\frac{3}{5})}{(\frac{3}{5})(1-\frac{1}{81})} = \ln\left(\frac{1}{120}\right) = \underline{-4.7875}$$

Similarly, we calculate the Odds Ratio values for all features in the 'business' category. This time we use values from Table 3.4:

$$OR(viagra, business) = \ln \frac{P(viagra|business)(1 - P(viagra|\overline{business}))}{P(viagra|\overline{business})(1 - P(viagra|business))}$$
$$= \ln \frac{\left(\frac{N_{F,C_k}}{N_{C_k}}\right)\left(1 - \frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)}{\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)\left(1 - \frac{N_{F,C_k}}{N_{C_k}}\right)} = \ln \frac{(\frac{0}{3})(1 - \frac{5}{6})}{(\frac{5}{6})(1 - \frac{0}{3})}$$

but when considering singularities:

$$= \ln \frac{(\frac{1}{N^2})(1 - \frac{1}{5})}{(\frac{5}{5})(1 - \frac{1}{N^2})} = \ln \frac{(\frac{1}{81})(\frac{1}{6})}{(\frac{5}{6})(\frac{80}{81})}$$
$$= \ln \frac{(\frac{1}{486})}{(\frac{400}{486})} = \ln\left(\frac{1}{400}\right) = \underline{-5.9915}$$

$$OR(save, business) = \ln \frac{(\frac{2}{3})(1 - \frac{1}{6})}{(\frac{1}{6})(1 - \frac{2}{3})} = \ln(10) = \underline{2.3026}$$

$$OR(erection, business) = \ln \frac{(\frac{1}{81})(\frac{1}{2})}{(\frac{1}{2})(\frac{80}{81})} = \ln\left(\frac{1}{80}\right) = \underline{-4.3820}$$

$$OR(cell, business) = \ln \frac{(\frac{2}{3})(\frac{80}{81})}{(\frac{1}{81})(\frac{1}{3})} = \ln(160) = \underline{5.0752}$$

$$OR(ski, business) = \ln \frac{(\frac{1}{3})(1 - \frac{2}{6})}{(\frac{2}{6})(1 - \frac{1}{3})} = \ln(1) = \underline{0.0}$$

Finally, we calculate the Odds Ratio values for all features in the 'private' category. This time we use values from Table 3.5:

$$OR(viagra, private) = \ln \frac{P(viagra|private)(1 - P(viagra|\overline{private}))}{P(viagra|\overline{private})(1 - P(viagra|private))}$$
$$= \ln \frac{\left(\frac{N_{F,C_k}}{N_{C_k}}\right)\left(1 - \frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)}{\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)\left(1 - \frac{N_{F,C_k}}{N_{C_k}}\right)} = \ln \frac{(\frac{1}{2})(1 - \frac{4}{7})}{(\frac{4}{7})(1 - \frac{1}{2})}$$
$$= \ln \frac{(\frac{1}{2})(\frac{3}{7})}{(\frac{4}{7})(\frac{1}{2})} = \ln \frac{(\frac{3}{7})}{(\frac{4}{7})} = \ln\left(\frac{3}{4}\right) = \underline{-0.2877}$$

$$OR(save, private) = \ln \frac{(\frac{1}{81})(\frac{4}{7})}{(\frac{3}{7})(\frac{80}{81})} = \ln\left(\frac{1}{60}\right) = \underline{-4.0943}$$

$$OR(erection, private) = \ln \frac{(\frac{1}{81})(\frac{4}{7})}{(\frac{3}{7})(\frac{80}{81})} = \ln\left(\frac{1}{60}\right) = \underline{-4.0943}$$

$$OR(cell, private) = \ln \frac{(\frac{1}{81})(\frac{5}{7})}{(\frac{2}{7})(\frac{80}{81})} = \ln\left(\frac{5}{160}\right) = \underline{-3.4657}$$

$$OR(ski, private) = \ln \frac{(\frac{80}{81})(\frac{6}{7})}{(\frac{1}{7})(\frac{1}{81})} = \ln(480) = \underline{7.1738}$$

Globalizing the Odds Ratio values Table 3.15 lists the Odds Ratio values (without logarithm) for all three categories in our example, and an aggregated

value (sum). This last column is one way of globalizing the Odds Ratio, that is, to create *one* ranked list of features for *all* categories. Odds ratio without the use of logarithm is not common for feature selection, but we show it for comparison reasons.

Term	Spam	Business	Private	Sum
ski	0.0083	1.0000	480.0000	481.0083
viagra	320.0000	0.0025	0.7500	320.7525
erection	240.0000	0.0125	0.0167	240.0292
cell	0.0188	160.0000	0.0313	160.0501
save	0.5000	10.0000	0.0167	10.5167

Table 3.15: E-mail example: Features ranked by summarized Odds Ratio values. Note that values in this table are without logarithm.

Table 3.16 shows the (log) Odds Ratio values from our e-mail example, and the sum of these values. This sum is called Extended Odds Ratio in [CHTQ09], and is computed as in Equation (3.14). Notice that this method of globalizing produced a differently ranked list than the one in Table 3.15.

$$EOR(F) = \sum_{k=1}^{|C|} OR(F, C_k) \tag{3.14}$$

Another way to globalize the Odds Ratio (or any other local feature selection

Term	ln(Spam)	ln(Business)	ln(Private)	Extended OR
ski	-4.7875	0.0000	6.1738	1.3863
viagra	5.7683	-5.9915	-0.2877	-0.5109
cell	-3.9766	5.0752	-3.4657	-2.3671
save	-0.6931	2.3026	-4.0943	-2.4848
erection	5.4806	-4.3820	-4.0943	-2.9957

Table 3.16: E-mail example: Features ranked by Extended Odds Ratio (EOR) values.

method) is suggested in [EM07, Chapter 14.2.7][4]. Here, the Odds Ratio value for a (feature, category) pair is multiplied by the probability for a document to receive that category label, before summarizing the values, as shown in Equation (3.15).

$$WOR(F) = \sum_{k=1}^{|C|} \frac{N_{C_k}}{N} \times OR(F, C_k) \tag{3.15}$$

This method is called Weighted Odds Ratio in [CHTQ09]. Notice that this method of globalizing produce a differently ranked list than both the one in Table 3.15 and the one in Table 3.16.

Note that these Odds Ratio method prefer positive features, that is, evidence that a document belongs to a category. The MOR and CDM metrics proposed

[4]Note that no log statement is used in the Odds Ratio equation in [EM07, Chapter 14.2.7]

Term	ln(Spam)	ln(Business)	ln(Private)	Weighted OR
viagra	5.7683	-5.9915	-0.2877	0.5026
erection	5.4806	-4.3820	-4.0943	0.0653
save	-0.6931	2.3026	-4.0943	-0.4504
ski	-4.7875	0.0000	6.1738	-0.7558
cell	-3.9766	5.0752	-3.4657	-0.8458

Table 3.17: E-mail example: Features ranked by Weighted Odds Ratio (WOR) values.

in [CHTQ09], variants of Odds Ratio, considers the absolute values of the log values when summarizing, and hence weights positive and negative evidence equally. The CDM metric is presented in Section 3.7.6.

We compared Weighted OR and Extended OR (in addition to OR without the logartihm statement) (Figure B.1). The Extended OR showed the best performance in our experiments, and hence this is used in the main comparison of feature selection methods in Chapter 4.

3.7.6 Class Discrimination Measure (CDM)

The Class Discrimination Measure (CDM) was presented in [CHTQ09] as a means for improving text categorization performance on multi-class data sets with Naïve Bayes. It was developed on the basis of Odds Ratio, and was shown to perform better than several other ways of extending Odds Ratio for multi-class situations. Also, it was shown to perform better than the well-known Information Gain metric.

The rather simple CMD feature selection method is shown in Equation (3.16). We handle singularities the same way as for Odds Ratio.

$$CDM(F) = \sum_{k=1}^{|C|} \left| \log \frac{P(F|C_k)}{P(F|\overline{C_k})} \right| = \sum_{k=1}^{|C|} \left| \log \frac{\left(\frac{N_{F,C_k}}{N_{C_k}}\right)}{\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)} \right| \tag{3.16}$$

Example

We demonstrate the CDM metric by calculating the value for the feature 'viagra' in the 'spam' category. We use $\log_2$ for this calculation, but the choice of logarithm base does not affect the ranging of the features, and is hence not

important.

$$
\begin{aligned}
CDM(viagra) &= \sum_{k=1}^{|C|} \left| \log \frac{\left(\frac{N_{F,C_k}}{N_{C_k}}\right)}{\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right)} \right| \\
&= \left| \log \frac{\left(\frac{4}{4}\right)}{\left(\frac{1}{5}\right)} \right| + \left| \log \frac{\left(\frac{0}{3}\right)}{\left(\frac{5}{6}\right)} \right| + \left| \log \frac{\left(\frac{1}{2}\right)}{\left(\frac{4}{7}\right)} \right| \\
&\text{but when considering singularities:} \\
&= \left| \log \frac{\left(1 - \frac{1}{N^2}\right)}{\left(\frac{1}{5}\right)} \right| + \left| \log \frac{\left(\frac{1}{N^2}\right)}{\left(\frac{5}{6}\right)} \right| + \left| \log \frac{\left(\frac{1}{2}\right)}{\left(\frac{4}{7}\right)} \right| \\
&= \left| \log \frac{\left(\frac{80}{81}\right)}{\left(\frac{1}{5}\right)} \right| + \left| \log \frac{\left(\frac{1}{81}\right)}{\left(\frac{5}{6}\right)} \right| + \left| \log \frac{\left(\frac{1}{2}\right)}{\left(\frac{4}{7}\right)} \right| \\
&= \left| \log \left(\frac{400}{81}\right) \right| + \left| \log \left(\frac{2}{135}\right) \right| + \left| \log \left(\frac{7}{8}\right) \right| \\
&= \left| \log \left(4.9383\right) \right| + \left| \log \left(0.0148\right) \right| + \left| \log \left(0.8750\right) \right| \\
&= \left| 2.3040 \right| + \left| 6.0768 \right| + \left| 0.1926 \right| \\
&= 2.3040 + 6.0768 + 0.1926 \\
&= \underline{8.5735}
\end{aligned}
$$

The rest of the CDM values are calculated in the same way. We list them in Table 3.18, ranked by the total CDM value for each feature.

Term	Spam	Business	Private	Sum
erection	5.9248	5.3399	5.1175	16.3821
cell	5.0179	5.7549	4.5325	15.3053
viagra	2.3040	6.0768	0.1926	8.5735
ski	5.6028	0.0000	2.7894	8.3923
save	0.6781	2.0000	5.1175	7.7955

Table 3.18: E-mail example: Features ranked by Class Discrimination Measure.

3.7.7 Chi Square (CHI)

Feature Selection by χ^2 testing [YP97, MRS08] is based on Pearson's χ^2 (chi square) test. The χ^2 test is often used to test the independence of two variables. The null-hypothesis is that the two variables are completely independent of each other. The higher value of the χ^2 test, the closer relationship the variables have.

In feature selection, the χ^2 test measures the independence of a feature and a category. The null-hypothesis here is that the feature and category are completely

independent, i.e. that the feature is useless for categorizing documents. The higher χ^2 value for a (feature, category) pair, the less independent they are. Hence, the features with the highest χ^2 values for a category should perform best for categorizing documents.

Equation

$$\chi^2(F, C_k) = \frac{N \times \left((N_{F,C_k} \times N_{\overline{F},\overline{C_k}}) - (N_{F,\overline{C_k}} \times N_{\overline{F},C_k})\right)^2}{N_F \times N_{\overline{F}} \times N_{C_k} \times N_{\overline{C_k}}} \tag{3.17}$$

Example

First, we calculate the χ^2 values for the spam category, using Equation (3.17) and N-values from Table 3.3:

$$\begin{aligned}
\chi^2(viagra, spam) &= \frac{N \times \left((N_{F,C_k} \times N_{\overline{F},\overline{C_k}}) - (N_{F,\overline{C_k}} \times N_{\overline{F},C_k})\right)^2}{N_F \times N_{\overline{F}} \times N_{C_k} \times N_{\overline{C_k}}} \\
&= \frac{9 \times \left((4 \times 4) - (1 \times 0)\right)^2}{5 \times 4 \times 4 \times 5} = \frac{9 \times (16-0)^2}{400} \\
&= \frac{2304}{400} = \frac{144}{25} = \underline{5.76} \\
\chi^2(save, spam) &= \frac{9 \times \left((1 \times 3) - (2 \times 3)\right)^2}{3 \times 6 \times 4 \times 5} = \frac{81}{360} = \frac{9}{40} = \underline{0.225} \\
\chi^2(erection, spam) &= \frac{9 \times \left((3 \times 5) - (0 \times 1)\right)^2}{3 \times 6 \times 4 \times 5} = \frac{2025}{360} = \frac{45}{8} = \underline{5.625} \\
\chi^2(cell, spam) &= \frac{9 \times \left((0 \times 3) - (2 \times 4)\right)^2}{2 \times 7 \times 4 \times 5} = \frac{576}{280} = \frac{72}{35} = \underline{2.0571} \\
\chi^2(ski, spam) &= \frac{9 \times \left((0 \times 2) - (3 \times 4)\right)^2}{3 \times 6 \times 4 \times 5} = \frac{1296}{360} = \frac{18}{5} = \underline{3.6}
\end{aligned}$$

Then the business-category values are calculated, using equation (3.17) and N-values from Table 3.4:

$$\begin{aligned}
\chi^2(viagra, business) &= \frac{9 \times \left((0 \times 1) - (5 \times 3)\right)^2}{5 \times 4 \times 3 \times 6} = \frac{2025}{360} = \frac{45}{8} = \underline{5.625} \\
\chi^2(save, business) &= \frac{9 \times \left((2 \times 5) - (1 \times 1)\right)^2}{3 \times 6 \times 3 \times 6} = \frac{729}{324} = \frac{9}{4} = \underline{2.25} \\
\chi^2(erection, business) &= \frac{9 \times \left((0 \times 3) - (3 \times 3)\right)^2}{3 \times 6 \times 3 \times 6} = \frac{729}{324} = \frac{9}{4} = \underline{2.25} \\
\chi^2(cell, business) &= \frac{9 \times \left((2 \times 6) - (0 \times 1)\right)^2}{2 \times 7 \times 3 \times 6} = \frac{1296}{252} = \frac{36}{7} = \underline{5.1428} \\
\chi^2(ski, business) &= \frac{9 \times \left((1 \times 4) - (2 \times 2)\right)^2}{3 \times 6 \times 3 \times 6} = \frac{0}{324} = \underline{0.0}
\end{aligned}$$

Finally the private-category values are calculated, using equation (3.17) and

N-values from Table 3.5:

$$\chi^2(viagra, private) = \frac{9 \times \big((1 \times 3) - (4 \times 1)\big)^2}{5 \times 4 \times 2 \times 7} = \frac{9}{280} = \underline{0.0321}$$

$$\chi^2(save, private) = \frac{9 \times \big((0 \times 4) - (3 \times 2)\big)^2}{3 \times 6 \times 2 \times 7} = \frac{324}{252} = \frac{9}{7} = \underline{1.2857}$$

$$\chi^2(erection, private) = \frac{9 \times \big((0 \times 4) - (3 \times 2)\big)^2}{3 \times 6 \times 2 \times 7} = \frac{324}{252} = \frac{9}{7} = \underline{1.2857}$$

$$\chi^2(cell, private) = \frac{9 \times \big((0 \times 5) - (2 \times 2)\big)^2}{2 \times 7 \times 2 \times 7} = \frac{144}{196} = \frac{36}{49} = \underline{0.7346}$$

$$\chi^2(ski, private) = \frac{9 \times \big((2 \times 6) - (1 \times 0)\big)^2}{3 \times 6 \times 2 \times 7} = \frac{1296}{252} = \frac{36}{7} = \underline{5.1429}$$

Then we aggregate these values to create one global ranked list of selected features. We do this like we did with Word Frequencies in Section 3.7.1 and Mutual Information in Section 3.7.3 by assigning weights to each category's value based on the number of documents that are labeled with that category in the training set. Hence we can call it a weighted average:

$$\begin{aligned}\chi^2(viagra) &= \sum_{k=1}^{|C|} \frac{N_{C_k}}{N} \chi^2(viagra, C_k) \\ &= \frac{4}{9} \times +5.76\frac{3}{9} \times 5.625 + \frac{2}{9} \times 2.6036 \\ &= \underline{5.0136}\end{aligned}$$

Term	Spam	Business	Private	Sum	wAvg.
viagra	5.7600	5.6250	0.0321	11.4171	4.4421
erection	5.6250	2.2500	1.2857	9.1607	3.5357
cell	2.0571	5.1428	0.7346	7.9345	2.7917
ski	3.6000	0.0000	5.1429	7.6429	2.7428
save	0.2250	2.2500	1.2857	3.7607	1.1357

Table 3.19: E-mail example: Features ranked by global χ^2 value.

3.7.8 NGL Coefficient

The NGL coefficient presented in [NGL97] is a variant of the Chi square metric. It was originally named a 'correlation coefficient', but we follow Sebastiani [Seb02] and name it 'NGL coefficient' after the last names of the inventors Ng, Goh, and Low.

The NGL coefficient looks only for evidence of positive class membership, while the chi square metric also selects evidence of negative class membership. Hence, it is called a 'one-sided' chi square metric in [NGL97]. In their experiments, it performed better than chi square. In [RS99] it was better than Odds Ratio and Mutual Information on some feature set sizes, and worse on other.

Equation

The NGL Coefficient is computed by Equation (3.18)

$$NGL(F, C_k) = \frac{\sqrt{N}(N_{F,C_k}N_{\overline{F},\overline{C_k}} - N_{F,\overline{C_k}}N_{\overline{F},C_k})}{\sqrt{N_F N_{\overline{F}} N_{C_k} N_{\overline{C_k}}}} \tag{3.18}$$

Example

The NGL feature values for the spam category are calculated as follows, using Equation (3.18) and values from Table 3.3.

$$\begin{aligned}
NGL(viagra, spam) &= \frac{\sqrt{N}(N_{F,C_k}N_{\overline{F},\overline{C_k}} - N_{F,\overline{C_k}}N_{\overline{F},C_k})}{\sqrt{N_F N_{\overline{F}} N_{C_k} N_{\overline{C_k}}}} \\
&= \frac{\sqrt{9}(4 \times 4 - 1 \times 0)}{\sqrt{5 \times 4 \times 4 \times 5}} \\
&= \frac{3 \times (16 - 0)}{\sqrt{400}} = \frac{48}{20} = \frac{12}{5} = \underline{2.4} \\
NGL(save, spam) &= \frac{\sqrt{9}(1 \times 3 - 2 \times 3)}{\sqrt{3 \times 6 \times 4 \times 5}} = \frac{-9}{\sqrt{360}} = \underline{-0.4743} \\
NGL(erection, spam) &= \frac{\sqrt{9}(3 \times 5 - 0 \times 1)}{\sqrt{3 \times 6 \times 4 \times 5}} = \frac{45}{\sqrt{360}} = \underline{2.3717} \\
NGL(cell, spam) &= \frac{\sqrt{9}(0 \times 3 - 2 \times 4)}{\sqrt{2 \times 7 \times 4 \times 5}} = \frac{-24}{\sqrt{280}} = \underline{-1.4343} \\
NGL(ski, spam) &= \frac{\sqrt{9}(0 \times 2 - 3 \times 4)}{\sqrt{3 \times 6 \times 4 \times 5}} = \frac{-36}{\sqrt{360}} = \underline{-1.8973}
\end{aligned}$$

We skip the computation details for the other categories, and present the feature values in Table 3.20. We aggregated the feature values by computing a weighted average as we also did for Word Frequency, Mutual Information and Chi Square.

Term	**Spam**	**Business**	**Private**	**Sum**	**wAvg.**
erection	2.3717	-1.5000	-1.1339	-0.2622	0.3021
viagra	2.4000	-2.3717	-0.1793	-0.1510	0.2362
save	-0.4743	1.5000	-1.1339	-0.1082	0.0372
cell	-1.4343	2.2678	-0.8571	-0.0236	-0.0720
ski	-1.8973	0.0000	2.2678	0.3705	-0.3393

Table 3.20: E-mail example: Features ranked by global NGL coefficient.

3.7.9 GSS Coefficient

The GSS coefficient was originally presented in [GSS00] as a 'simplified chi square function'. We follow [Seb02] and name it GSS after the names on the inventors

Galavotti, Sebastiani, and Simi.

$$GSS(F, C_k) = N_{F,C_k} N_{\overline{F},\overline{C_k}} - N_{F,\overline{C_k}} N_{\overline{F},C_k} \tag{3.19}$$

The experiments in [GSS00] showed far better results when using *max* as a globalizing strategy rather than average, hence we follow them on that:

$$GSS(F) = \max_{k=1}^{|C|} GSS(F, C_k) \tag{3.20}$$

Example

We show the calculation of the GSS coefficient with the feature 'viagra' in the 'spam' category, using Equation (3.19) and values from Table 3.3.

$$\begin{aligned}
GSS(viagra, spam) &= N_{F,C_k} N_{\overline{F},\overline{C_k}} - N_{F,\overline{C_k}} N_{\overline{F},C_k} \\
&= (4.0 \times 4.0) - (1.0 \times 0.0) \\
&= 16.0 - 0.0 = \underline{16.0} \\
GSS(viagra, business) &= (0.0 \times 1.0) - (5.0 \times 3.0) \\
&= 0.0 - 15.0 = \underline{-15.0} \\
GSS(viagra, private) &= (1.0 \times 3.0) - (4.0 \times 1.0) \\
&= 3.0 - 4.0 = \underline{1.0}
\end{aligned}$$

We then use the maximum as a global value for the 'viagra' feature in all categories:

$$\begin{aligned}
GSS(viagra) &= \max_{k=1}^{|C|} GSS(viagra, C_k) \\
&= \max GSS(viagra, \{spam, business, private\}) \\
&= \max\{16.0, -15.0, 1.0\} = \underline{16.0}
\end{aligned}$$

The rest of the example values are computed similarly, and we present them in Table 3.21.

Term	**Spam**	**Business**	**Private**	**Max**
viagra	16.0	-15.0	1.0	16.0
erection	15.0	-9.0	-6.0	15.0
cell	-8.0	12.0	-4.0	12.0
ski	-12.0	0.0	12.0	12.0
save	-3.0	9.0	-6.0	9.0

Table 3.21: E-mail example: Features ranked by GSS coefficient.

3.7.10 Bi-Normal Separation (BNS)

Bi-Normal Separation [For03, For02] is defined by Equation (3.21). If we model the occurrence of some feature in each document by the event of a random Normal variable exceeding a hypothetical threshold, the prevalence rate of that feature corresponds to the area under the curve beyond the threshold. If the feature is more prevalent in one category than the others, then its threshold is further from the tail of the curve than that of the other categories, and vice versa. The BNS method measures the distance or *separation* between these thresholds. Using absolute values, it captures the

The BNS equation as presented in [For03]:

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

where F^{-1} is the standard Normal distribution's inverse cumulative probability function. Translating to our own notation, the BNS equation looks like this:

$$BNS(F, C_k) = \left| F^{-1}\left(\frac{N_{F,C_k}}{N_{C_k}}\right) - F^{-1}\left(\frac{N_{F,\overline{C_k}}}{N_{\overline{C_k}}}\right) \right| \tag{3.21}$$

where again F^{-1} is the standard Normal distribution's inverse cumulative probability function. Since this function explodes if $N_{F,C_k}/N_{C_k} = 0$ or $N_{F,\overline{C_k}}/N_{\overline{C_k}} = 0$, we follow Forman and set a minimum rate to $5/10000$. Likewise, we set the maximum rate to $1 - 5/10000$.

When aggregating these values we find the global BNS value for each feature F. A pilot study we ran using 25, 50, 75 and 100 features showed best results by using a weighted average (as we did for WF, MI, Chi Square and NGL), so we do that in the experiments presented in Chapter 4.

$$BNS(F) = \sum_{k=1}^{|C|} \frac{N_{C_k}}{N} BNS(F, C_k) \tag{3.22}$$

Example

The e-mail example is ranked by weighted average BNS values in Table 3.22. We show here the calculation of BNS value for the 'viagra' feature in the 'spam' category, using equations (3.21) and values from Table 3.3.

$$\begin{aligned} BNS(viagra, spam) &= \left| F^{-1}\left(\frac{4}{4}\right) - F^{-1}\left(\frac{4}{4}\right) \right| \\ &= |F^{-1}(1.0) - F^{-1}(1.0)| \end{aligned}$$

Since reaching the upper limit, we substitute 1.0 with the minimum value 0.0005:

$$\begin{aligned} BNS(viagra, spam) &= |F^{-1}(0.0005) - F^{-1}(0.0005)| \\ &= |3.2905 - -0.8416| \\ &= |4.1322| = \underline{4.1322} \end{aligned}$$

The rest of the values are calculated in the same way, and are presented in Table 3.22.

Term	Spam	Business	Private	wAvg
erection	3.9650	3.2905	3.1105	3.5503
viagra	4.1321	4.2579	0.1800	3.2958
cell	3.0372	3.7213	2.7246	3.1957
ski	3.5439	0.0000	4.3581	2.5435
save	0.4211	1.3981	3.1105	1.3444

Table 3.22: E-mail example: Features ranked by weighted average Bi-Normal Separation value.

3.7.11 Categorical Proportional Difference (CPD)

One of the latest additions to the feature selection family is the Categorical Proportional Difference or CPD [SH08]. It considers only positive examples from the training data.

Equations

Equation (3.23) shows how to compute the feature value for a feature, category pair, while Equation (3.24) shows how to use the highest value as the global feature value, as recommended in [SH08].

$$CPD(F, C_k) = \frac{N_{F,C_k} - N_{F,\overline{C_k}}}{N_F} \tag{3.23}$$

$$CPD(F) = \max_k\{CPD(F, C_k)\} \tag{3.24}$$

Background

Categorical Proportional Difference is in [SH08] reported to have excellent performance both for Naïve Bayes and Support Vector Machines, but at the cost of low aggressivity levels. Their study used an exhaustive search to find the percentage of features each feature selection method performed best at. When keeping 61.2 to 76.5% of the features, CPD performed better than Odds Ratio, Information Gain, and several others. However, except for Mutual Information, all other methods had their best performance at a much higher aggressivity level.

Example

We show the computation of the feature values for the e-mail example, using equations (3.23) and (3.24).

$$CPD(viagra, spam) = \frac{N_{F,C_k} - N_{F,\overline{C_k}}}{N_F} = \frac{4-1}{5} = 0.6$$
$$CPD(viagra, business) = \frac{N_{F,C_k} - N_{F,\overline{C_k}}}{N_F} = \frac{0-5}{5} = -1.0$$
$$CPD(viagra, private) = \frac{N_{F,C_k} - N_{F,\overline{C_k}}}{N_F} = \frac{1-4}{5} = -0.6$$
$$CPD(viagra) = \max\{0.6, -1.0, -0.6\} = \underline{0.6}$$
$$CPD(save, spam) = \frac{1-2}{3} = -0.3333$$
$$CPD(save, business) = \frac{2-1}{3} = 0.3333$$
$$CPD(save, private) = \frac{0-3}{3} = -1.0$$
$$CPD(save) = \max\{-0.3333, 0.3333, -1.0\} = \underline{0.3333}$$

We skip the computation details for the remaining features, and present the feature values in Table 3.23.

Term	**Spam**	**Business**	**Private**	**Max**
erection	1.0000	-1.0000	-1.0000	1.0000
cell	-1.0000	1.0000	-1.0000	1.0000
viagra	0.6000	-1,0000	-0.6000	0.6000
save	-0.3333	0.3333	-1.0000	0.3333
ski	-1.0000	-0.3333	0.3333	0.3333

Table 3.23: E-mail example: Features ranked by Category Proportional Difference.

3.7.12 DIA Association Factor

The Darmstadt Indexing Approach (DIA) is used in the AIR/X system – a rule-based system for indexing with terms from a predescribed vocabulary [FHK+91]. The DIA (Darmstadt Indexing Approach) Association factor is calculated with Equation (3.25).

$$z(F, C_k) = P(C_k|F) = \frac{P(F \cap C_k)}{P(F)} = \frac{\frac{N_{F,C_k}}{N}}{\frac{N_F}{N}} = \frac{N_{F,C_k}}{N_F} \tag{3.25}$$

Notice how this equation also equals WF/DF, that is, the feature selection method Word Frequency explained in Section 3.7.1 divided by the other method Document Frequency explained in Section 3.6.4. As such, it can be seen as a hybrid of two methods.

While the DIA considers a much wider set of features than us (they consider *properties* of terms, documents, categories and pairwise relationships among these), our evaluation of the DIA association factor is as any other feature selection method. That is, while we evaluate the DIA association factor, we do so using regular single features as in the rest of the experiments in this report.

Example

We go through the e-mail example with this method as well. We start with the spam category, using equation (3.25) and values from Table 3.3.

$$z(viagra, spam) = \frac{N_{F,C_k}}{N_F} = \frac{4}{5} = \underline{0.8}$$
$$z(save, spam) = \frac{N_{F,C_k}}{N_F} = \frac{1}{3} = \underline{0.3333}$$
$$z(erection, spam) = \frac{N_{F,C_k}}{N_F} = \frac{3}{3} = \underline{1.0}$$
$$z(cell, spam) = \frac{N_{F,C_k}}{N_F} = \frac{0}{2} = \underline{0.0}$$
$$z(ski, spam) = \frac{N_{F,C_k}}{N_F} = \frac{0}{3} = \underline{0.0}$$

The feature values for the other categories are computed similarly, so we skip the details and present the results in Table 3.24. We aggregate these values like we have done with several other methods by weighting the results for each category by the percentage of documents that category contains in our training set. We denote these global values weighted average, wAvg.

Term	Spam	Business	Private	Sum	wAvg.
erection	1.0000	0.0000	0.0000	1.0	0.4444
viagra	0.8000	0.0000	0.2000	1.0	0.4000
save	0.3333	0.6667	0.0000	1.0	0.3702
cell	0.0000	1.0000	0.0000	1.0	0.3333
ski	0.0000	0.3333	0.6667	1.0	0.2593

Table 3.24: E-mail example: Features ranked by global DIA Association factor

3.8 Untested Methods

Although we have tried to include as many feature selection metrics as possible, and probably have compared more metrics in one system than ever published before, our comparison is by no means complete. The following methods are not tested in our system, because of time issues. There are probably several other methods in use as well.

The MOR metric presented in [CHTQ09], several variants of the Relief algorithm can be found in [YH08], Count Difference in [CS08], the Posterior Inclusion Probability (PIP) in [EM07], and the Pearson product-moment correlation in [GLM07].

In [DDH+07], the two unsupervised methods Subspace Sampling (SS) and Weight-based Sampling (WS) are presented, in [ZZH04] the MC-OR odds ratio based method for multi-class problems is proposed, while [For03] evaluates (Log) Probability Ratio, F_1-measure, Odds Ratio Numerator, Accuracy, Accuracy balanced, and Power, in addition to some of the methods we have tested.

[Mla98] proposed ExpP, FreqOddsRatio, and FreqLogP, [KS97] used the Expected Cross Entropy, [KYMW97] presented the Keyword Checker, while the Gain Ratio metric was proposed in [Qui86].

Chapter 4

Experiments and Results

In this section, we first present some text collections commonly used for text categorization, including the 20 Newsgroups that we used in our experiments. We then describe how we conducted the text categorization experiments, including hardware and software environment, and settings used in the generation of vector files to represent the collection and how we used Weka.

4.1 Text Categorizing Collections

In this section we present some of the text collections available for evaluation text categorization systems. For the '20-Newsgroups' collection we used in our experiments, we describe how we prepared it, and how our use of it differs from earlier work.

4.1.1 Reuters-21578

Reuters-21578 is the most widely used text categorization test collection. It contains 21,578 documents (28.0 MB) that appeared on the Reuters newswire in 1987. The Reuters-21578 collection is used in [SH08, CHTQ09, XWLJ08, For03, CMS01, Joa98]. Earlier versions of the collection include the Reuters-22173 released in January 1993, which was used in [YP97, LR94, Lew92].

Reuters-21578 consists of five category sets – 'EXCHANGES', 'ORGS', 'PEOPLE', 'PLACES', and 'TOPICS'. The 'TOPICS' category set is used in most text categorization research, and we will use this set as well. It consists of 135 categories, including 'coconut', 'gold', 'inventories', and 'money-supply'. The categories are hierarchically organized, and hence the document distribution is highly skewed. 120 of the categories were assigned to at least one document, leaving 15 unused/empty categories. Of the 120 used categories, 57 were assigned to 20 or more documents. Some documents concern several topics, and the text categorization used in this collection is thus the multiple label type, as discussed in Section 2.1.

4.1.2 Reuters Corpus Volume 1 (RCV1)

The RCV1 corpus [LYRL04] was released by Reuters Ltd. in 2000, succeeding the popular 'Reuters-21578' text categorization test collection. RCV1 contains about

810,000 Reuters, English language News stories[1] (about 2.5 GB) from August 20, 1996 to August 19, 1997. One advantage with this collection over the 21578 is of course the much larger size. RCV1 is used in [BMGMF08, CS08, DDH+07].

4.1.3 20-Newsgroups

The 20 Newsgroups collection is a collection of about 20,000 messages posted in 1993 to 20 Usenet newsgroups. They were gathered by Tom Mitchell's graduate student Ken Lang while he was working on [Lan95]. For text categorization, we use the newsgroups as categories, and the messages as documents.

The newsgroups, listed in Table 4.1, each contain around 1,000 messages. Some groups are rather similar, like the 'comp.sys.ibm.pc.hardware' and the 'comp.sys.mac.hardware' groups, while others are not, for instance 'rec.autos' and 'alt.atheism'.

The original collection contains header information that reveals which group or groups a message belongs to, for instance 'Newsgroups: rec.sport.baseball'. It is obviously important to remove this information, not doing so would give unrealistic results. The original collection also contains around 4% cross-postings (messages posted to more than one newsgroup). We make use of a version provided by Jason Rennie[2] that contains the 'From:' and 'Subject:' lines of the headers, and where duplicates are removed. As such, it represents a single label text categorization problem. It contains 18828 messages.

Jason Rennie also has a version with a pre-defined split based on publication dates, which would be a natural choice if using a single training/test split. Since we are using cross-validation however, the '18828' version was a better alternative.

comp.graphics	rec.autos	sci.crypt
comp.os.ms-windows.misc	rec.motorcycles	sci.electronics
comp.sys.ibm.pc.hardware	rec.sport.baseball	sci.med
comp.sys.mac.hardware	rec.sport.hockey	sci.space
comp.windows.x		
misc.forsale	talk.politics.misc	talk.religion.misc
	talk.politics.guns	alt.atheism
	talk.politics.mideast	soc.religion.christian

Table 4.1: The 20 Newsgroups collection. The messages are almost evenly distributed in the newsgroups

Dasgupta et al. [DDH+07] used five of Rennie's ten test-train splits. They compared three sampling-based feature selection techniques to Mutual Information (they named it Information Gain) and Document Frequency. They found that Mutual Information outperformed the other measures, especially with an aggressive selection strategy.

Simeon and Hilderman [SH08] split the 20-Newsgroup collection into ten unique sub-problems with two or three categories in each set. They performed 10-fold cross-validation in Weka within each subset and averaged their results.

[1] There is also a multilingual corpus called RCV2

[2] The '18828' version, available at `http://people.csail.mit.edu/jrennie/20Newsgroups/`.

They performed an exhaustive search to find the maximum possible F_1-measure, where a classifier was trained and tested with every top-ranked subset (the one best feature, the two best features, and so on up until all features were used) from a ranked and sorted feature list – within each split of the dataset. This way, they found the exact number of features for each selection method where that method performed the best. They then ranked the feature selection methods according to classifier performance, resulting in a list with very different feature set sizes for each method. Using an SVM classifier for instance, they ranked their own CPD method on top for the 20-Newsgroups collection with 76.5% features kept, while Odds Ratio was rated number 4 with only 35.8% kept.

We handled the 20 Newsgroups collection in a different manner than both [DDH+07] and [SH08]. This means that we do not expect to see the exact same results as they did. A notable difference is at the tokenization level. While Rennie used the Rainbow program for tokenization, we used Lucene and its StandardTokenizer. Rennie's tokenization removed all punctuations and all numbers. StandardTokenizer recognizes e-mail addresses, time and dates. For instance, we have the e-mail address '10326.97.uupcb@ compdyn.questor.org' in our feature list, while Rennie's feature list contains 'uupcb', 'compdyn' and so on. Also, we have '12:00:00 AM', they have 'am'. Further, Rennie did not employ stemming, which we do. And finally, Rennie did not make use of any stop word list, while we make use of Lucene's standard English stop word list in addition to the removal of some individually selected features.

Our preprocessing of the 20 Newsgroups collection reduced it from 155195 to 53730 features, from which we selected feature sets of 500 to 10000 (roughly 0.93 to 18.61 percent) features. While it is obvious from this that we will see other results than [DDH+07] with so many differences, note that we compare the feature selection methods against each other with the same pre-processing done for every method. Hence, our results should be just as valuable as those in for instance [DDH+07], but the results should not be directly compared. The results are relative to each other.

4.1.4 The OHSUMED Test Collection

The OHSUMED test collection [HBLH94] contains a subset of the MEDLINE database. It contains 348,566 references (about 400 MB of data) covering all references from 270 medical journals over a five year period (1987-1991). The collection is originally an information retrieval text collection, but it has been used for text categorization research by several, including [SH08, Wit08, XWLJ08, LTSL07, For03, HK00, RS99, Joa98, YP97].

Not all the references in the collection contain abstracts, some contain just headings. Only those with abstracts are relevant for us, as the abstract is the the 'document' or text to be categorized. An example of an OHSUMED document can be found in Appendix A.2.

The OHSUMED collection was indexed with MeSH[3] terms. The MeSH headings are hierarchically structured. There are around 18,000 categories in MeSH, 14,321 are present in the OHSUMED collection. The main headings

[3]Medical Subject Headings (MeSH) is a controlled vocabulary medical thesaurus developed by the National Library of Medicine. See `http://www.nlm.nih.gov/mesh/` for details.

are 'Anatomy [A]', 'Organisms [B]', 'Diseases [C]' etc. – 16 in total[4]. The 'Diseases [C]' heading for instance, contains 23 subheadings, including 'Bacterial Infections and Mycoses [C01]' and 'Virus Diseases [C02]', which themselves contain subheadings and so forth. These headings can be used as category labels when using the collection for text categorization research.

Yang and Pedersen [YP97] used the documents from 1990 as training documents, and the 1991 documents for testing. Their 1990 training set had a total of 72,076 unique terms, and an average of 12 category labels assigned to each document. Hence, they used the collection as a multiple label collection. They further used the heading combined with the abstract as the document, and there is no mention of the removal of heading-only documents.

Joachims [Joa98] used the first 10,000 of the 50,216 documents from 1991 which have abstracts as training documents and the second 10,000 as testing documents. They further used only the 23 MeSH 'Diseases [C]' categories, and flattened the hierarchical structure, so that any document indexed under a subheading of 'Virus Diseases [C02]' were regarded as belonging to the C02 category. They used the collection as a multi-label problem, and treated each (overlapping) category as an independent binary classification problem. Their OHSUMED selection and split was later used by [SH08, LTSL07].

Han and Karypis [HK00] prepared 15 sets with 10 classes and around 1,000 documents each. Only documents with abstracts were selected. Documents with multiple MeSH topics within each set were discarded. This way, they extracted a collection with non-overlapping categories (aka. single label) from the original multiple-label collection. Only four of these sets were used in [HK00]. These four sets were later also used by [For03].

We followed Han and Karypis, and prepared the OHSUMED collection as close to their partitioning as possible. Using their four published topic sets, we extracted document sets with slightly fewer documents than their collection. Unfortunately, we did not have enough computers available to include this corpus in our comparison experiments in addition to the 20 Newsgroups corpus.

4.2 Experiment Setup

In this section we describe how we prepared for and conducted our experiments, including the hardware and software environment in the first section, and the use of Weka in Section 4.2.2. We also explain a simple scheme we used for performing combination experiments in Section 4.2.3.

4.2.1 Hardware and Software

Our experiments were run on two standard (IBM PC) computers with 13-16 GB RAM and 64-bit Intel(R) Xeon(R) CPUs (E5430) running at 2.66GHz. Both were installed with Ubuntu 8.10.

Our own implementation used indexing, stemming, tokenization etc. from Lucene[5] version 2.4.0. We used the well-known data mining software Weka-3-6-0[6]

[4] The 2009 MeSH tree structure can be browsed at `http://www.nlm.nih.gov/cgi/mesh/2009/MB_cgi`

[5] `http://lucene.apache.org/java/docs/`

[6] `http://www.cs.waikato.ac.nz/ml/weka/`

for creating and evaluating classifiers. Two learners were used, both with the standard Weka settings:

- Naive Bayes: weka.classifiers.bayes.NaiveBayes
- Support Vector Machine: weka.classifiers.functions.SMO

The Java[7] version used was 1.6.0_10.

4.2.2 Categorization Experiments

In this section, we explain how we generated vector files with the various feature selection methods, and conducted the categorization experiments using Weka with these files.

First we indexed the collections using our own software based on Lucene:

```
$ ./run.sh no.ntnu.idi.tm.collectionmanager.CollectionIndexer \
   -c 20news-18828 -s true
```

where we specified the corpus (`-c`), and used stemming (`-s`). A simple shell script was used for setting java options including classpath and memory space.

Next, we ran each feature selection method for each of the six number of features, and exported the results to `arff.gz` files:

```
$ ./run.sh no.ntnu.idi.tm.collectionmanager.CollectionExporter \
   -c 20news-18828 -t 2 -m INFORMATION_GAIN -n 500
```

where we specified the corpus (`-c`), a minimum of two characters for each feature (`-c 2`), the feature selection method (`-m`), and the number of features to be selected (`-n`). Also, not shown here is the default value to remove features with a document frequency of one.

Next, Weka was used to create and evaluate learners. We started Weka 3.6.0 in Experimenter mode [BFH+08, Chapter 5], added a classifier (e.g. Naïve Bayes), added a set of vector files (e.g. information gain vector files, one for each of the 6 numbers of features), and specified 10-fold cross validation. This experiment was then saved in `.xml` format. We also made some edits to these files, including changing the name and location of the output file to something resembling the actual experiment, and correcting some path differences between the client and the server.

Next, the `.xml` files were copied to one of our two servers for processing. Here, we ran the Weka experiments with commands like the following example:

```
$ java weka.experiment.Experiment \
   -l naive-bayes_INFORMATION_GAIN.xml -r
```

where `-l` means load experiment from file and `-r` means run experiment. This step actually builds a classifier using the set of vector files (the selected features), evaluates its performance, and stores the results in another `.arff` file. This file contains one result line for each run, fold and dataset. Hence in our case, the result files from the NB experiments contained 600 result lines: 6 feature set sizes (from 500 to 10000 features) times 10 folds times 10 runs (each fold is run

[7] `http://java.sun.com/`

10 times). The result files from the SVM experiments contained 150 result lines: 3 feature set sizes (500, 1000 and 2000 features) times 10 folds times 5 runs.

The results files from Weka were then analyzed in the Weka Experimenter Analyzer window. Here, we ran the appropriate tests (e.g. F-measure, recall etc.) with our preferred output format gnuplot[8]. These tests average the results and normally present one value for each dataset (in our case one value for each feature selection method). The results from the various tests can be seen in Section 4.3.

4.2.3 Combination Experiments

We performed a range of NB classification experiments where we combined two or three feature selection methods. This was partly motivated by the fact that several high performing feature selection metrics selected very diverse feature sets, as can be seen in Table 4.3. For the combination of the feature sets from two feature selection methods, we used a rather simple algorithm:

We first specified to use `n` features from each `m` feature selection method. Each method was then used individually to rank the features. Then, the topmost ranked features for all methods were added to the combined set of features, if they were not already there. Furthermore, the next best features from each list were handled in the same way, and so on. This continued until `n` times `m` features were added to the new set.

Now, `n` or more features from the new set also occurred in each of the original `m` ranked lists. It might (and in most cases will) be more than `n` features because more than one method might have selected the same feature.

For instance, let's assume we specified `-n 2` and `-m INFORMATION_GAIN:TFDF` (two methods). Then the combined feature list would contain 4 features. If the two methods ranked the same four features on top, these four would be in the combined set, and the combination would be equal to each of the two. If the two methods had none of the same features in their two top-ranked positions, the algorithm would stop after investigating only these two times two features, as it would have filled up all its own four feature positions.

Vector files of the 20 Newsgroups collection were generated like the following example. Here, the two methods TFDF and CHI were combined, producing a vector file where each document was represented using a total of 500 features, where at least 250 of these were among the top ranked features of each method's ranked list.

```
$ ./run.sh no.ntnu.idi.tm.collectionmanager.CollectionExporter \
   -c 20news-18828 -t 2 -m TFDF:CHI_SQUARE -n 250
```

Our combinaion scheme is different from that used in [RY02]. There, the weights of each feature were normalized, and the maximum was chosen from the two.

4.3 Results and Discussion

We have conducted a series of experiments, comparing 17 feature selection methods on Naïve Bayes and Support Vector Machine classifiers. In this section,

[8] `http://www.gnuplot.info/`

we present and discuss the results obtained from these experiments. Eight main graph plots are presented, namely percent correctly classified documents, macroaverage precision, recall and F_1-measure, for the NB classifier and the SVM classifier (Figures 4.1 through 4.8[9]).

In Section 4.3.1 we discuss some general observations from the experiments, including variations and similarities between the two classifier types, groups of feature selections with similar performances, and our results compared to those of others that also used the 20 Newsgroups. We also discuss some observations of precision and recall results, and the results of the unsupervised feature selection methods against those of the supervised methods. Then in Section 4.3.2 we go through each feature selection method in turn.

We also performed a series of combination experiments. Their results are discussed in Section 4.3.3, where we also present a correlation matrix showing the similarities of the methods we evaluated.

4.3.1 General Observations

In this section we discuss some general observations from the experiment results.

The Classifier Matters

As expected, the Support Vector Machine (SVM) classifier performed much better than the Naïve Bayes (NB) classifier. While the best method (Chi Square) peaks at 69,19 percent correctly classified documents on NB (with a standard deviation[10] of 0.96), the highest recorded percentage on SVM (also Chi Square) was 84.89 (with a standard deviation of 0.73).

While SVM in general performs better than NB, it also uses considerably longer time. Table 4.2 shows the CPU times for the experiments run using the weirdness factor for feature selection. As an example, consider the set of 2000 features. With 10 folds averaged over 10 runs, this means 100 passes over the dataset (100 times the values shown in the table). For NB, 12.34 seconds times 100 is 1234 seconds, or about 21 minutes. For SVM, 732.98 seconds times 100 is 73298 seconds, or more than 20 hours. That is about 60 times as long as the NB classifier.

It is evident that the choice of classifier matters, and that performance needs should be considered together with needs for efficiency.

Result Groups

Most of the results from the high performance methods seem to be grouped into three partitions. For the NB classifier, both for percentage correct and F-measure, the top performing group includes CHI, GSS, EOR, IG and MI. These are all supervised methods. The next group includes the three unsupervised methods TFDF, CFIDF, and CF. The third group contains WF, a supervised method, and DF which is unsupervised.

The BNS method does not follow the performance curves of the mentioned groups. It performs excellent at the smallest feature set sizes, but does not

[9] While standard deviations are not shown in Figures 4.1 through 4.8, they are listed in Tables B.1 through B.8. Standard deviations are explained in Appendix B.1

[10] Standard deviations measure the variability of the results, as explained in Appendix B.1

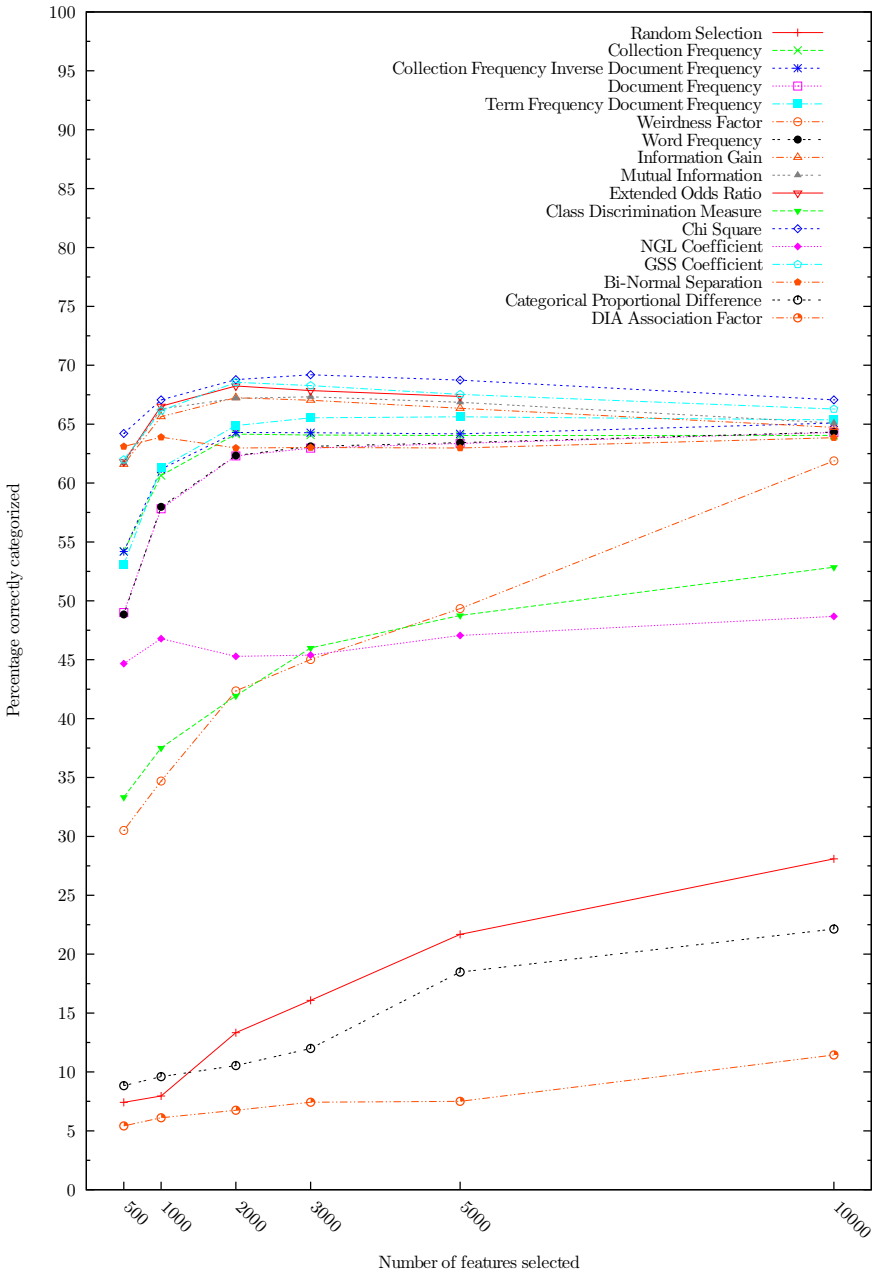

Figure 4.1: Percentage of documents correctly classified by a Naïve Bayes classifier using several feature selection methods at various numbers of features selected from the 20 Newsgroups collection

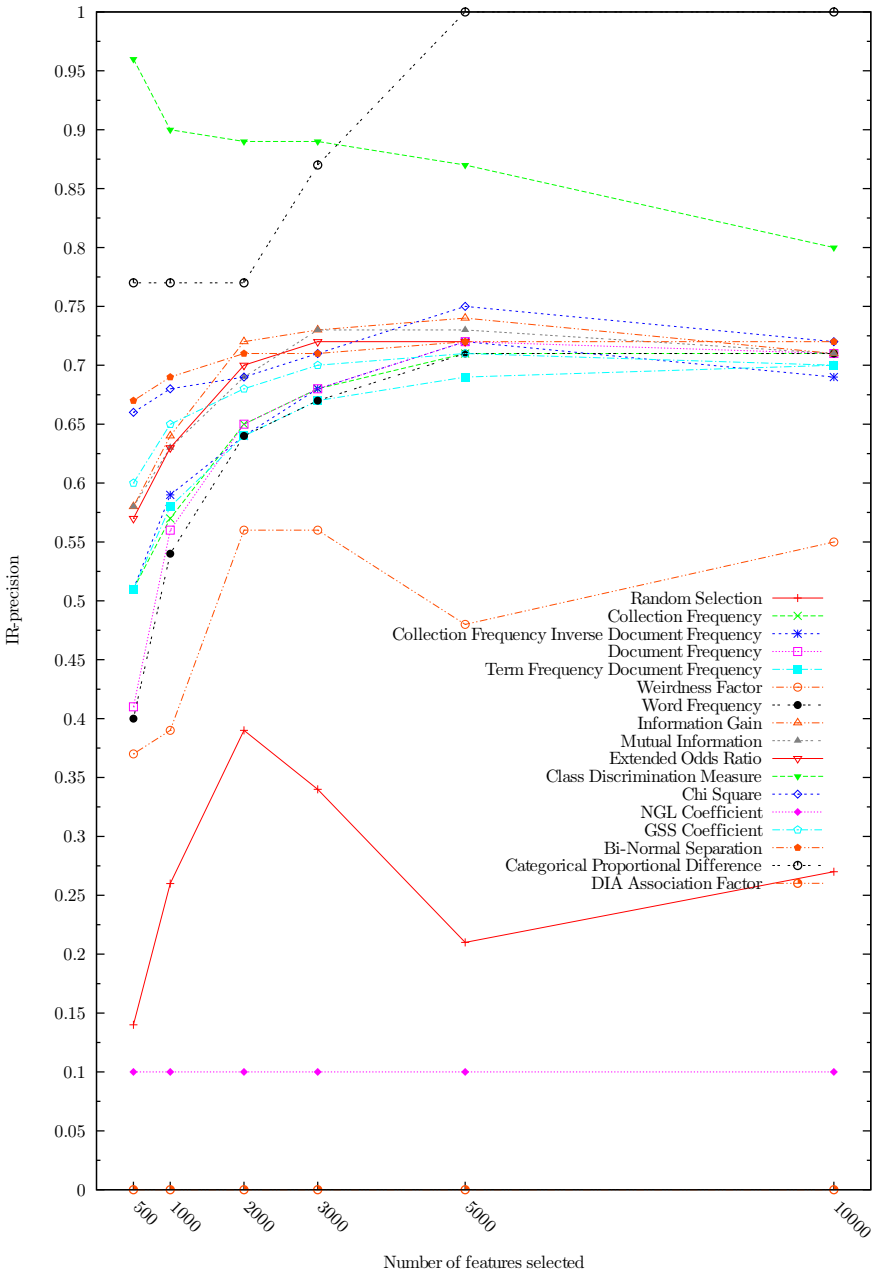

Figure 4.2: Macroaverage precision for a Naïve Bayes classifier using various feature selection methods at various numbers of features selected from the 20 Newsgroups collection

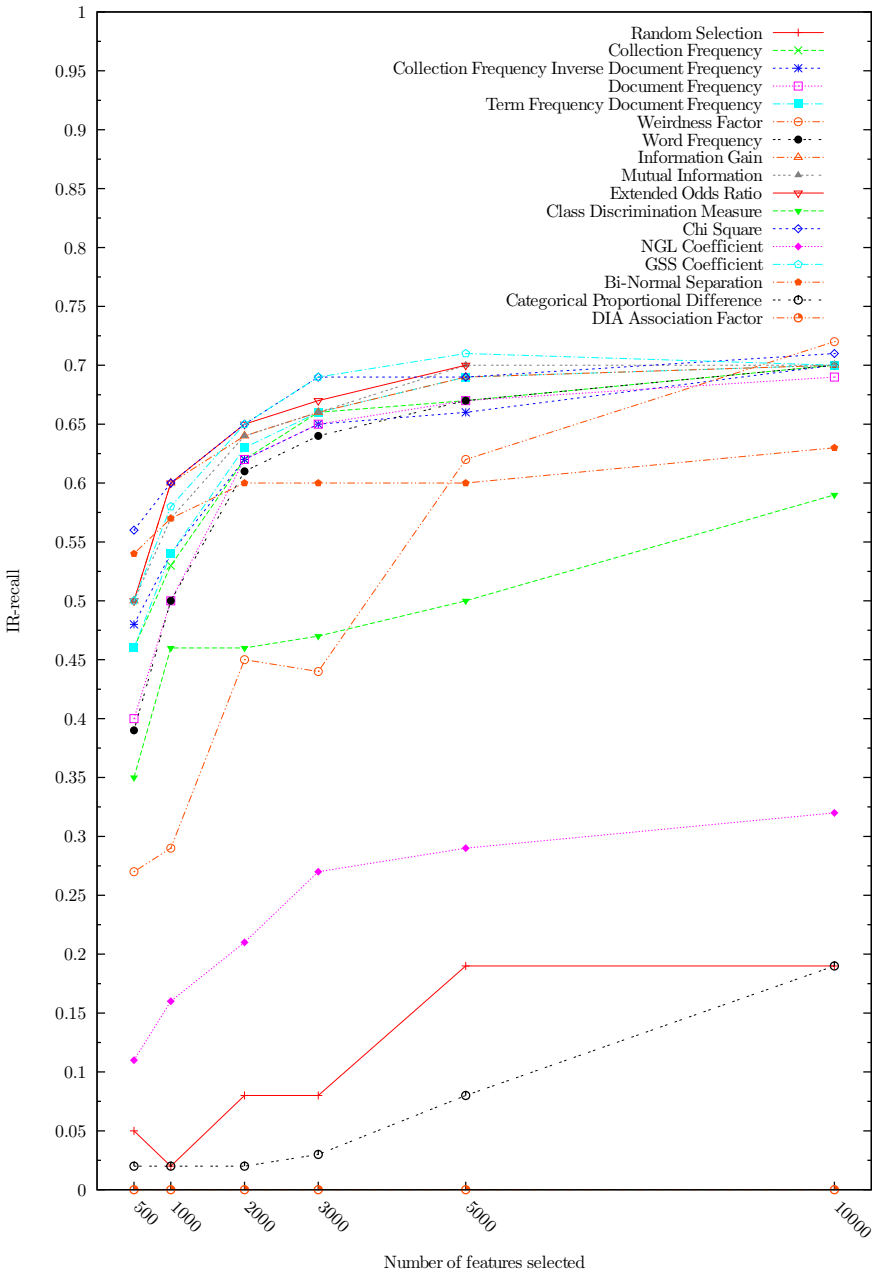

Figure 4.3: Macroaverage recall for a Naïve Bayes classifier using various feature selection methods at various numbers of features selected from the 20 Newsgroups collection

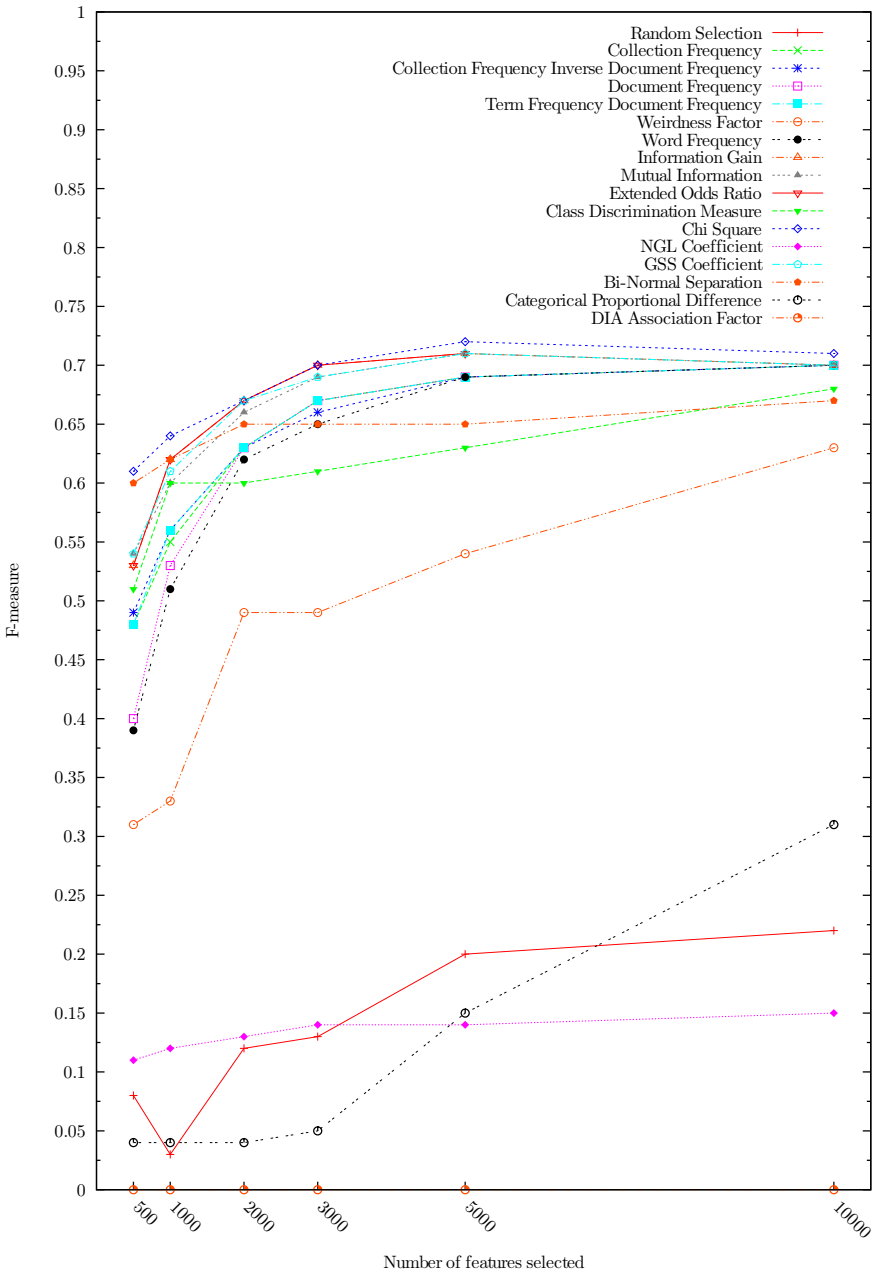

Figure 4.4: Macroaverage F_1-measure for a Naïve Bayes classifier using various feature selection methods at various numbers of features selected from the 20 Newsgroups collection

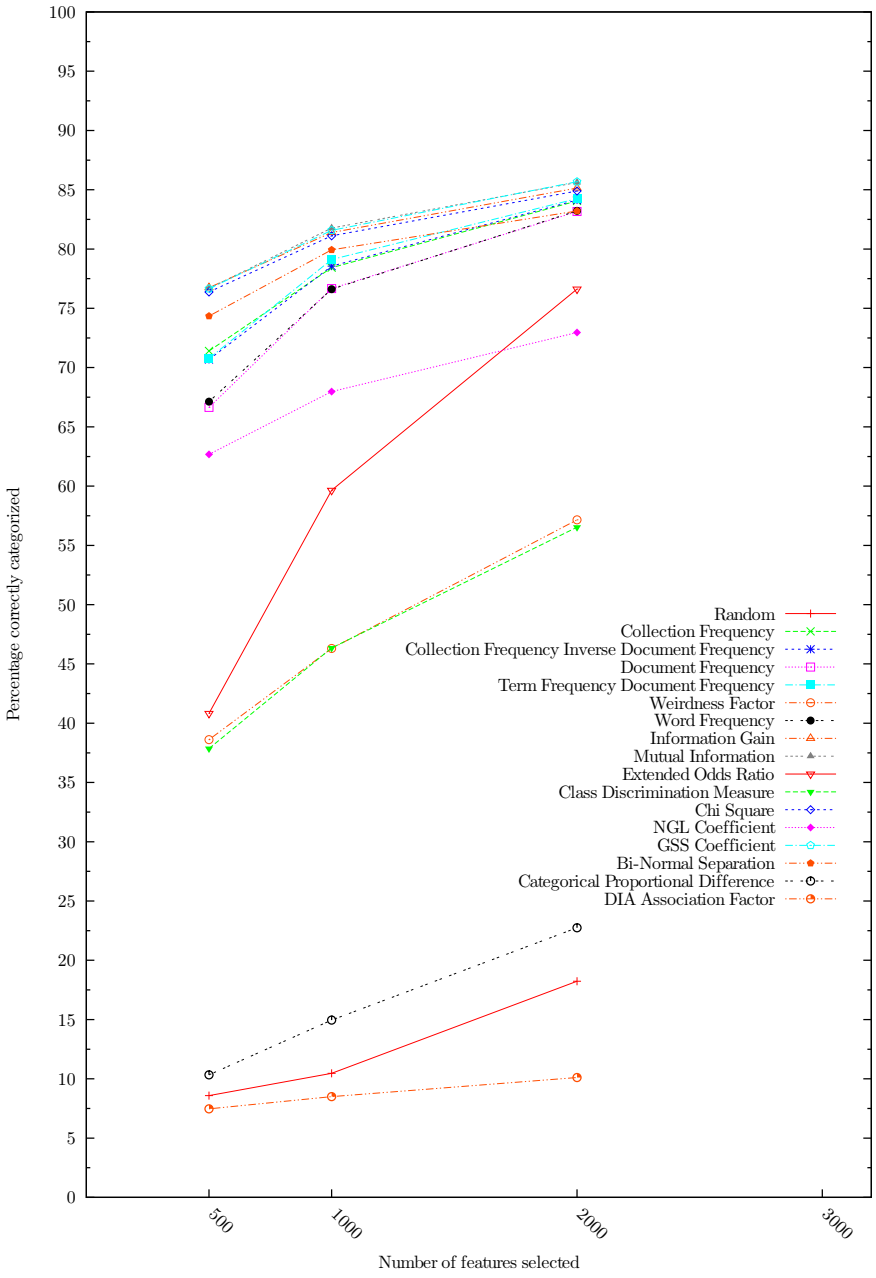

Figure 4.5: Percentage of documents correctly classified by a SVM classifier using various feature selection methods at various numbers of features selected from the 20 Newsgroups collection

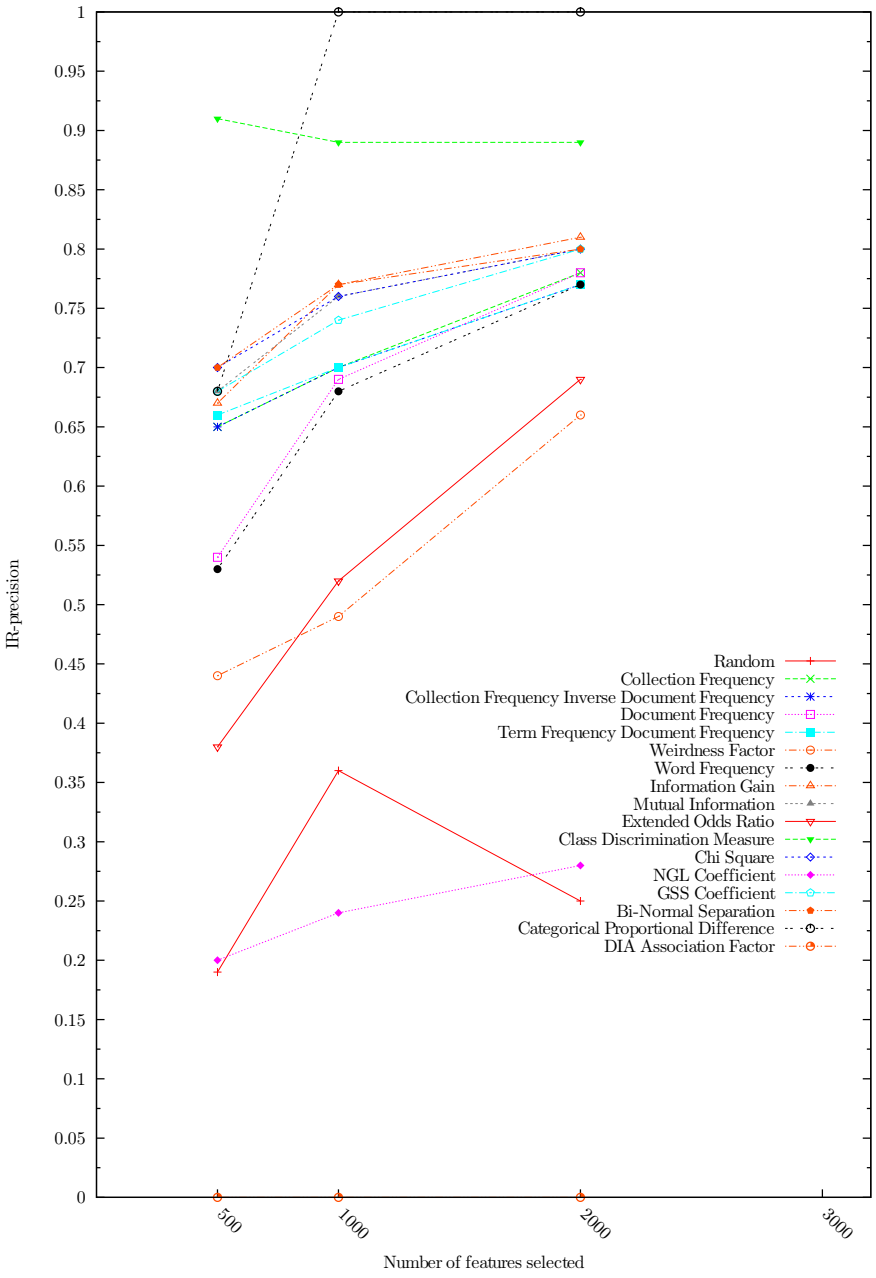

Figure 4.6: Macroaverage precision for a SVM classifier using various feature selection methods at various numbers of features selected from the 20 Newsgroups collection

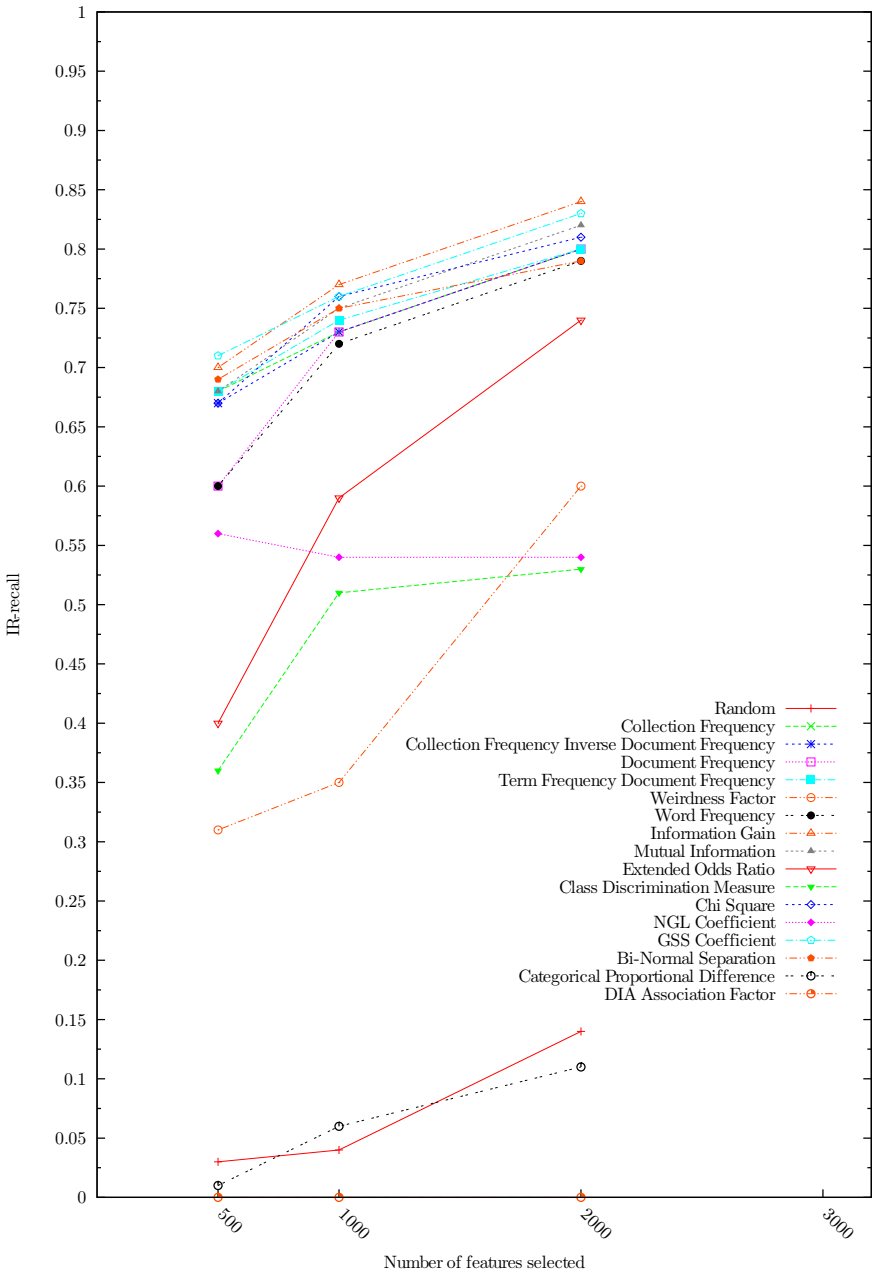

Figure 4.7: Macroaverage recall for a SVM classifier using various feature selection methods at various numbers of features selected from the 20 Newsgroups collection

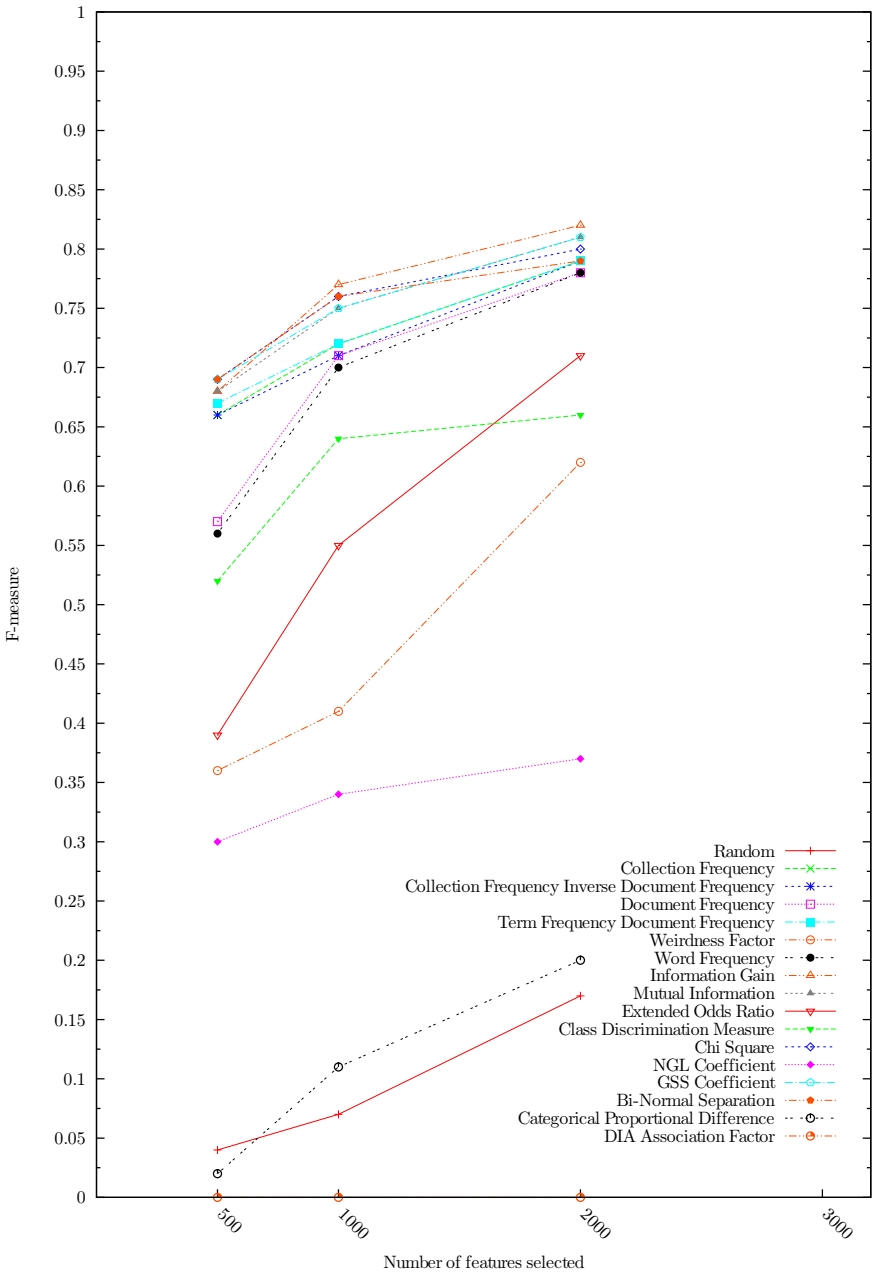

Figure 4.8: Macroaverage F_1-measure for a SVM classifier using various feature selection methods at various numbers of features selected from the 20 Newsgroups collection

Classifier: Dataset - Feature set size	User CPU time
NB: WEIRDNESS-500	2.40 ± 0.28
NB: WEIRDNESS-1000	5.24 ± 0.54
NB: WEIRDNESS-2000	12.34 ± 2.07
NB: WEIRDNESS-3000	17.98 ± 0.36
NB: WEIRDNESS-5000	40.31 ± 0.80
NB: WEIRDNESS-10000	130.41 ± 4.09
SVM: WEIRDNESS-500	168.25 ± 1.07
SVM: WEIRDNESS-1000	330.48 ± 4.78
SVM: WEIRDNESS-2000	732.98 ± 16.19

Table 4.2: CPU time comparison for Naïve Bayes (NB) and Support Vector Machine (SVM) classifiers. All experiments were run using 10-fold cross-validation, averaged over 10 runs. The values shown are average times for one pass over the data.

continue this trend. The remaining methods show more diverse results, and are not correlated to each other.

For the SVM charts, the situation is not far from that of the NB classifier. While SVM performs better in general, most methods seem to have the same impact on the SVM classifier as they have on NB. The top group of feature selection methods includes CHI, GSS, IG and MI here as well. Only EOR is gone from the top performing group. BNS again has a performance curve slightly different from the other high performing methods, showing excellent performance with the highest aggressivity level. The three unsupervised methods TFDF, CF and CFIDF is again following the first group closely, and WF and DF again forms a third group.

Some of the methods we have tested are ranked in [Seb02] (their rank is based on others' results). While experiments conducted by different researchers never can be compared directly, we make some notes here: Firstly, our results agree with their rank in that EOR and GSS are among the top performers. However, their top-ranked methods include NGL_{sum}. In our results, NGL performs poorly in all tests. It should be noted that we used a sum weighted by the class size, while they report the sum only. Other factors can also be different in the experiments, including test collection and the number of classes. Furthermore, in our experiments, CHI was always among the top performers, while this was not the case in their rank.

Our Results Compared to Others

The 20 Newsgroups collection was also used with NB and SVM classifiers in [SH08]. Their results show F-measure values higher than 0.91 on NB and 0.94 on SVM for all methods they tested (including CHI, IG, OR and CPD). As such, their results are much higher than our results. One reason for the differences, is that they divided the collections into subtasks with two or three categories in each task. Obviously, classification is easier when selecting between two or three categories than when selecting one of 20, like we do. Moreover, as explained in Section 4.1.3 their results are presented with rather large and varying feature set

sizes, while we evaluate the classifiers at fixed, more aggressive feature set sizes.

[DDH+07] also conducted experiments on the 20 Newsgroups collection. However, they used another classifier (Regularized Least Squares Classification) so their results cannot be compared directly to ours. Nevertheless, they find that IG performs better than DF, and that both IG and DF performs better than random feature selection. These results are in line with our results. The other feature selection methods we compare are not evaluated there, and vice versa.

Precision and Recall

The precision values are slightly higher than the recall values in our tests (see Section 2.5 for a discussion of the evaluation metrics including precision and recall). That is, most methods have up to 0.1 point higher precision than recall value. Some methods have significant differences however. For instance, the CPD metric has extremely high precision values (0.77 to 1.0 for NB), while at the same time worse recall values than random (0.02 to 0.2 points for NB). The CDM metric in fact shows a degrading (but very high) precision curve as the feature set size increases, while the recall curve increases.

A third interesting precision-recall difference is that of the NGL coefficient. While the percentage correctly classified documents stays rather unchanging on NB (around 45 to 50 %) as the feature set size increases, the precision curve stays flat on 0.1 points, while the recall rises from 0.1 to 0.32. On SVM on the other hand, recall falls while precision rises, as more features are used.

Unsupervised and Supervised Methods

As expected, supervised feature selection metrics are topping the charts in our comparison. But more interesting, the three unsupervised methods TFDF, CF and CFIDF performed better than the supervised WF method. Also, the three are not very far behind the leading cluster of methods. While starting at about 5 to 10% lower than the best methods in the percentage correct chart (Figures 4.1 and 4.5), they are just a few percent behind at 3000 features and above.

4.3.2 Individual Results

Below we discuss the results of each of the feature selection methods. We discuss the methods in the order they appear in the report, including the order they are listed in the main result charts (Figures 4.1 through 4.8).

Collection Frequency (CF)

The collection frequency metric performs very good both on the NB and SVM classifiers. Moreover, it shows equally good performance on precision and recall. Together with CFIDF and TFDF, CF is the best unsupervised feature selection metric among the ones we compared.

Collection Frequency Inverse Document Frequency (CFIDF)

Like CF, CFIDF seems to have very good performance allover, both for NB and SVM. It has no surprises in the precision and recall results. Somewhat surprisingly, the CF method overall performed as good as the CFIDF, suggesting

that the IDF-weighting in CFDF is of little value. This is confirmed by the correlation matrix in Table 4.3, which shows that CF and CFIDF selected 97% of the same features at a feature set size of 5000.

Document Frequency (DF)

The document frequency does not compete among the very best methods in our tests, at least not at the most aggressive levels. That said, it shows good results on average, and like CF and CFIDF, there are no surprises in the precision and recall results. In our 5000-features experiments, DF selected 86% and 88% of the same features as CFIDF and CF, respectively.

Term Frequency Document Frequency (TFDF)

TFDF shows very good results, and along with CF and CFIDF it is in the second best group of methods in our experiments. Looking at the correlation matrix in Table 4.3, we see that TFDF selected 91% of the same features as both CF and CFIDF at 5000 features, explaining the correlated performance results. While TFDF only is compared to DF in [XWLJ08], and only precision results are presented there, our experiments confirmed their findings: TFDF performs better than DF in almost all our experiments. Like in their results, the difference is most prominent in the most aggressive selection level.

Weirdness Factor

The weirdness factor does generally not show performance results among the best methods, but shows a steep curve, meaning it could possibly be better than the other methods for lower accuracy. Also, the best recall number we found, was from using the weirdness factor and a feature set size of 10000. We believe that while obviously not the best ranking function, weirdness factor might be a good way of filtering out noisy features.

The weirdness factor has been used for IR tasks, but not for feature selection for text categorization before. Our experimental results show that it might have a future here as well. While not among the best methods, it performs several times better than random feature selection. Moreover, with 10000 features on NB, it produced the highest recall results among all metrics, while correctly classifying a decent 62% of the documents.

Interestingly, the correlation matrix in Table 4.3 shows that the weirdness factor was less than 20% correlated with all other methods at 5000 feature selected, suggesting that other methods select more common features.

Word Frequency (WF)

The word frequency shows stable and good results is all our evaluations: it is consistently in the third best group of feature selection methods. One interesting observation, is that the supervised WF method seems to be highly correlated with the unsupervised DF method. In fact, they selected 98% of the same features at 5000 features. This suggests that the extra effort of considering the category information as done in WF adds little value to the document frequencies alone.

Information Gain (IG)

The well-known IG metric shows excellent performance in all our experiments, being among the top performers for both NB and SVM. Also, it shows stable results for both precision and recall. It is between 75% and 78% correlated with the unsupervised metrics (except weirdness factor) at 5000 features. It is 95% correlated with MI, showing its close relationship with this method.

Mutual Information (MI)

As mentioned, MI performs very closely to IG. They are both in the best group of metric in all our experiments.

Extended Odds Ratio (EOR)

The EOR metric had excellent performance when used with NB. It was in the best group of methods, and showed stable, reliable results for both precision and recall. For the SVM classifier on the other hand, it is surprisingly not among the top performers at all.

Class Discrimination Measure (CDM)

The CDM metric did not show impressing results in our experiments. In percentage correctly classified documents, CDM performed closely to that or the weirdness factor. While this is several times better than random selection, it is far from the best performing group. An interesting observation is that CDM shows very high precision levels - high above the best group of methods. Furthermore, its precision is highest at the smallest feature set size. At the same time, its recall values are much lower than the best group of methods. As such, it can be said to sacrifice completeness for a higher level of exactness. In the combined F-measure, CDM performs considerably better than the weirdness factor, close to the best methods.

Chi Square (CHI)

CHI was one of the best feature selection method all over in our experiments. For NB, it persistently classified the highest percentage of documents correctly, and it showed the best F-measure of all methods. For SVM, it also performed excellently, being in the topmost group at all feature set sizes. In Table 4.3, we see that CHI never shared more than 80% of the features with any other method at a set size of 5000. Most notably, CHI and EOR share only 36% of the features, while their results (at least for NB) should suggest otherwise.

NGL Coefficient

The NGL coefficient did not perform well in our tests, but produced some interesting results. In percentage correctly classified documents, it managed between 45% and 50% on all feature set sizes on NB. That is, hardly any increase at all when adding more features. Its precision curve stays even more stable, with 0.1 precision points for every feature set size. This number is far lower than we achieved using random feature selection. The recall score of NGL showed an increase from 0.1 to 0.32 as more feature were added. As such, opposite of the

CDM metric, it can be said to sacrifice exactness for completeness. But as both its scores are far below most other methods, it can not be recommended. In the combined F-measure, its performance was very poor, even worse than random selection at 5000 and 10000 features. Our poor NGL results are surprising, as the NGL metric was reported better than CHI in [NGL97].

Bi-Normal Separation (BNS)

The BNS metric showed very good results, but did not follow the result curves of the three groups of good metrics. while among the very best methods at the smallest feature sets, its performance did not increase much as much as that of the other methods with more features added. Its precision values are excellent, for high aggressivity levels, it had the best achieved among all the methods that managed to classify a decent percentage of the documents correctly. The fact that it performed so well with few features, but did not get much better with more features, suggests that it might be ranking its features better than the other methods. As many applications may need to prioritize a fast classification over a more correct classification, the BNS metric might look promising. For such cases, experiments with even smaller feature sets should be considered conducted.

The different performance curves of BNS compared to the other methods is confirmed by the correlation matrix in Table 4.3. At 5000 features, BNS never selected more than 33% of the same features as any other method.

Our results do not resemble those presented in [For03], but this was not expected either, as they used other corpora than we did. Moreover, they evaluate binary experiments and we conducted multi-class experiments.

Categorical Proportional Difference (CPD)

The Categorical Proportional Difference method shows poor results in our experiments. While its precision scores were extremely high (1.0 for 5000 and 10000 features), this is not valuable as it classified only from 7 to 22 percent of the documents correctly. However, it has a rather steep rising curve on the F-measure chart, suggesting that better results could be achieved with (much) larger feature set sizes, as those shown in [SH08]. As such, it looks more of a method for filtering out noisy features. It would be very interesting to see the performance of other methods if used *after* CPD, that is, first running CPD and keeping 60-80% of the features, and then more aggressively selecting features among the remaining ones using some other method.

While [SH08] reported F-measure values of more than 0.9 for the CPD metric with a Naive Bayes classifier on the 20 Newsgroups, we got F-measures of 0.04 to 0.31. This is partly because we used more aggressive feature selection (retaining only from less than 1% to about 19% of the features). But also the way we use the collection is important. Our classifier must choose between all 20 categories, while they split the corpus into partitions containing two or three categories each. Naturally, classifying documents into one of two or three categories is an easier task than one of 20, hence their results are better – for all feature selection methods.

The correlation matrix in Table 4.3 reveals that the CPD metric is very different from the other methods we compared. While selecting 14% of the same

features as DIA, and 7% of the same as the weirdness factor, the correlation with the other methods was between 1% and 3%. One fact should be noted about the correlation: In our experiments, the CPD metric assigned feature values of 1.0 to 23750 features, i.e. to 44.2% of the features. Moreover, it assigned 0.0 to 19942 features (37.1%). The remaining 10038 features received values in the range of <0,1>. Now, the question arises of how to rank the top 23750 features that received the same value. In our implementation, these were ranked randomly (or in fact as they appear in the Lucene index). As such, the correlation matrix does not provide reliable figures, it just shows the correlation in our actual implementation and index.

Nevertheless, these numbers suggest that there is a need for another way of ranking the top features. They also serve as a possible indication of why [SH08] needed over 70% of the features to achieve top scores for the 20 Newsgroups collection on NB.

DIA Association Factor

In all our experiments, the DIA association factor achieved the worst results of all methods. Surprisingly, it even performed worse than random selection. As explained in Section 3.7.12, this method was not proposed as a method of feature selection for the kind of single term features we consider here. Now that we have evaluated it, we can conclude that it is not usable in situations like those we present here.

4.3.3 Combination Results

We performed a range of NB classification experiments where we combined two or three feature selection methods, as explained in Section 4.2.3. This was partly motivated by the fact that several high performing feature selection metrics selected very diverse feature sets, as can be seen in Table 4.3.

Much of the conducted combination experiments did not lead to considerably better classification results. We present here only the combinations that showed good results, and place the rest of the figures in Appendix B.3. Each combination experiment is presented by two figures: The percentage of correctly classified documents, and the F-measure.

The two supervised, top-performing metrics CHI and GSS were combined, producing the results shown in Figures 4.9 and 4.10. As the GSS coefficient is based on Chi Square, one could be leaded to think of them as bad candidates for combination. The correlation matrix in Table 4.3 however, showed a correlation of 79% at 5000 features. When they both performed excellent, this suggested that there might be possible to get even higher performance from them combined. So was also the case the figures show. The combined effort achieved a higher percentage of correctly categorized documents than both methods for 500 to 3000 features. While CHI was the best single method, the CHI+GSS combination had the best results of all NB classifiers evaluated, measured in percentage correctly classified documents.

The F-measure was generally not better for the combination than for the best method, but at 3000 features it was. This was also the aggressivity level where the highest percentage of the features were categorized correctly.

Method	CF	CFIDF	DF	TFDF	Weird	WF	IG	MI	EOR	CDM	CHI	NGL	GSS	BNS	CPD
CFIDF	97	-													
DF	88	86	-												
TFDF	91	91	87	-											
Weird	9	10	7	9	-										
WF	89	86	98	87	7	-									
IG	76	76	75	78	12	75	-								
MI	74	73	72	75	13	72	95	-							
EOR	72	70	78	71	3	78	**55**	**52**	-						
CDM	**6**	**7**	**4**	**6**	19	**4**	**15**	**18**	**0**	-					
CHI	58	58	54	59	16	54	76	80	36	**33**	-				
NGL	34	35	31	36	19	32	41	44	24	27	48	-			
GSS	75	75	74	77	12	74	91	92	55	17	79	42	-		
BNS	**31**	**31**	**32**	**31**	11	**32**	**33**	**33**	**16**	**5**	**32**	15	33	-	
CPD	0	1	0	1	7	0	1	1	0	3	1	1	1	1	-
DIA	1	1	1	1	12	1	4	5	0	20	9	19	5	6	14

Table 4.3: Correlation of the evaluated feature selection methods when 5000 features are selected from the 20-Newsgroups collection. The percentage of the features selected that also were selected by the other methods are shown. The random selection method is not shown, as its selected features vary from time to time.

We wanted to combine the 'outsider' BNS with some of the best unsupervised methods. BNS and CFIDF achieved a considerably higher percentage of correctly classified documents at all feature set sizes except 500 (Figure 4.11) when combined than individually. The F-measure on the other hand (Figure 4.12) was not as convincing, the most aggressive selection levels were lower than the best method, while equal or better on the less aggressive levels.

BNS combined with TFDF achieved similar results as BNS and CFIDF. BNS was 31% correlated to both CFIDF and TFDF at 5000 features, while CFIDF and TFDF were 91% correlated.

Figures 4.15 and 4.16 show the results of combining the BNS and the CDM metric. The motivation for selecting these two metrics for combination experiments was that they were only 5 percent correlated at 5000 selected features, while at the same time they both show good or ok classification results. The two combined achieved a higher percentage of correctly classified documents, and also a higher F-measure score than each of the two alone. However, the combined result is still not as good as the best single metrics.

BNS also did well with IG, as depicted in Figures 4.17 and 4.18. The percentage of correctly classified documents were increased at all levels except 500, where it was equal to the best. The F-measure was not better than the best single metric at any aggressivity level. however.

BNS also lifted the performance of the GSS coefficient slightly. This is interesting, as the GSS factor was among the very best performers, and the two were only 33% correlated at 5000 features.

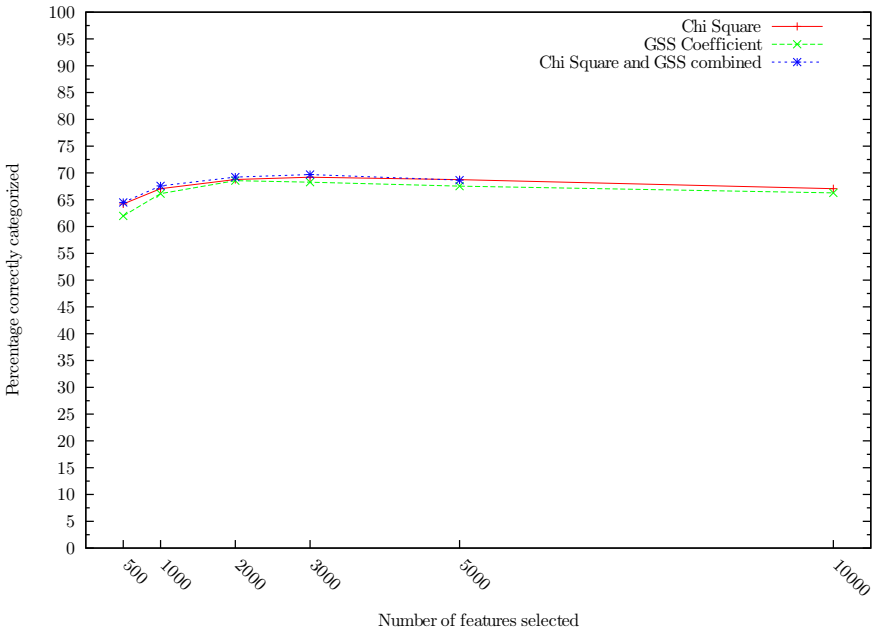

Figure 4.9: Percentage of documents correctly classified by an NB classifier using CHI, GSS, and the two combined on the 20 Newsgroups.

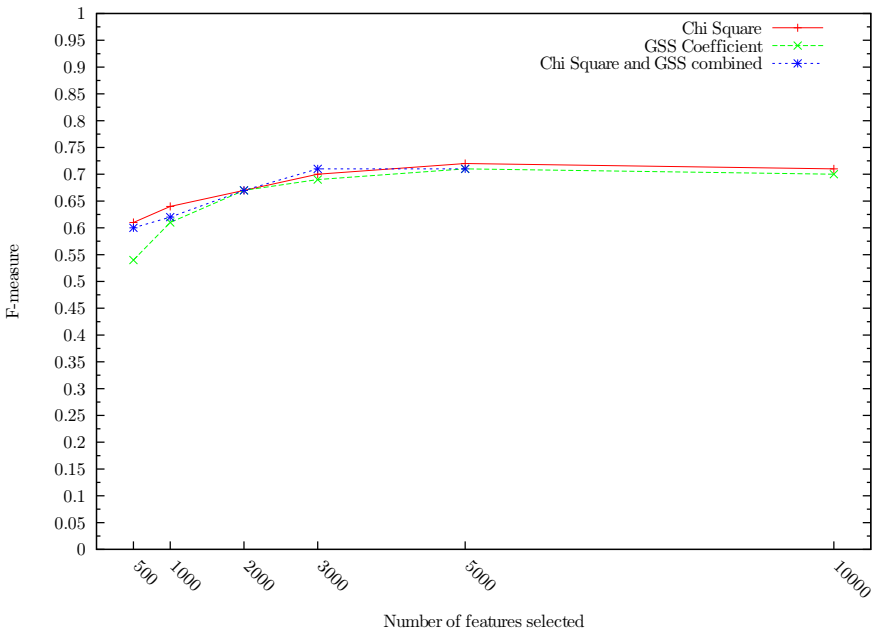

Figure 4.10: F-measure for a Naïve Bayes classifier using CHI, GSS, and the two combined at various numbers of features selected from the 20 Newsgroups.

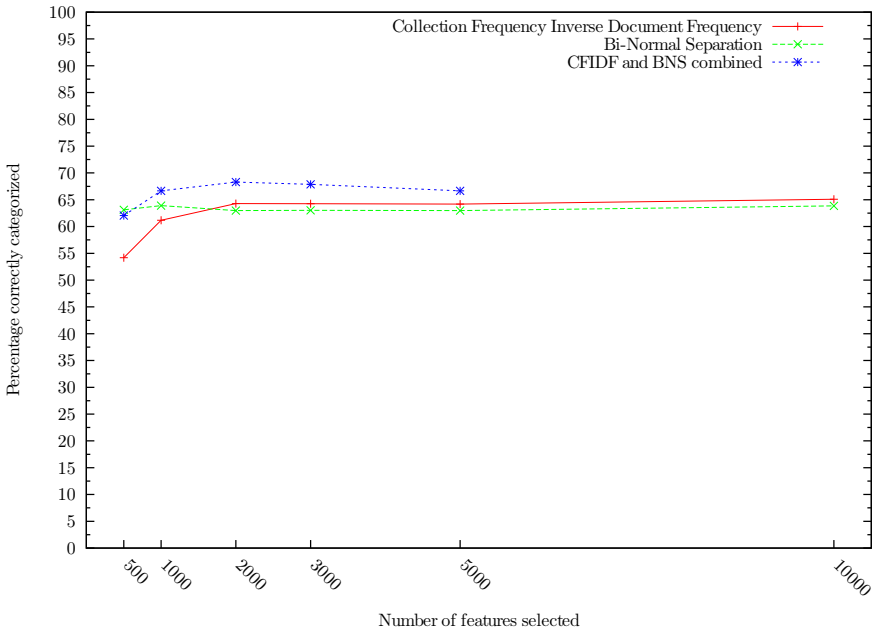

Figure 4.11: Percentage of documents correctly classified by an NB classifier using CFIDF, BNS, and the two combined on the 20 Newsgroups.

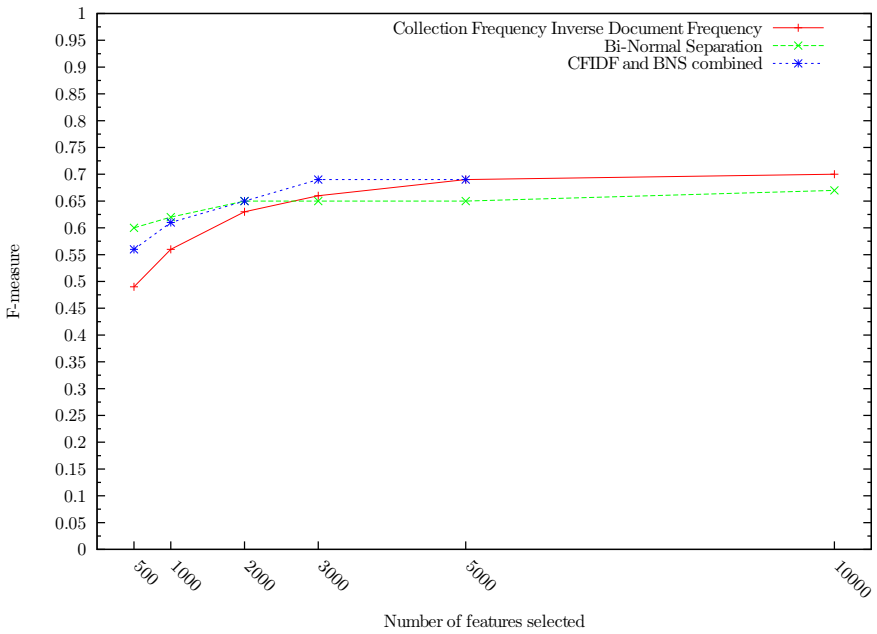

Figure 4.12: F-measure for a Naïve Bayes classifier using CFIDF, BNS, and the two combined at various numbers of features selected from the 20 Newsgroups.

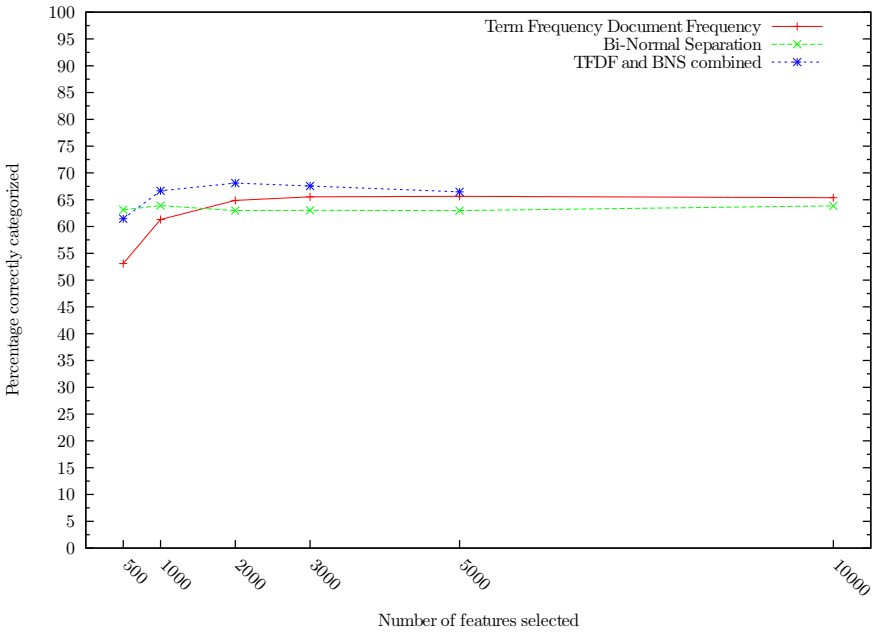

Figure 4.13: Percentage of documents correctly classified by a Naïve Bayes classifier using TFDF, BNS, and the two combined on the 20 Newsgroups.

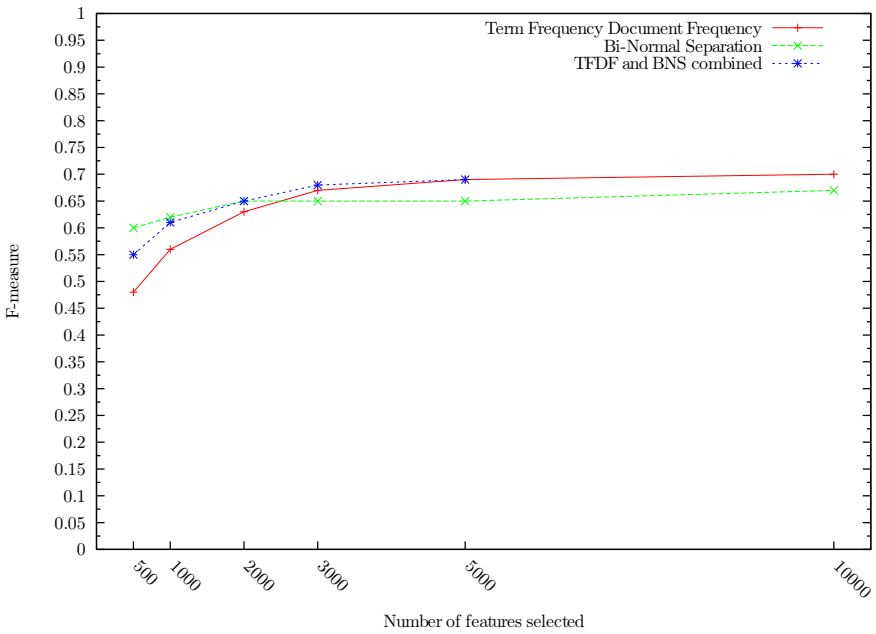

Figure 4.14: F-measure for a Naïve Bayes classifier using TFDF, BNS, and the two combined at various numbers of features selected from the 20 Newsgroups.

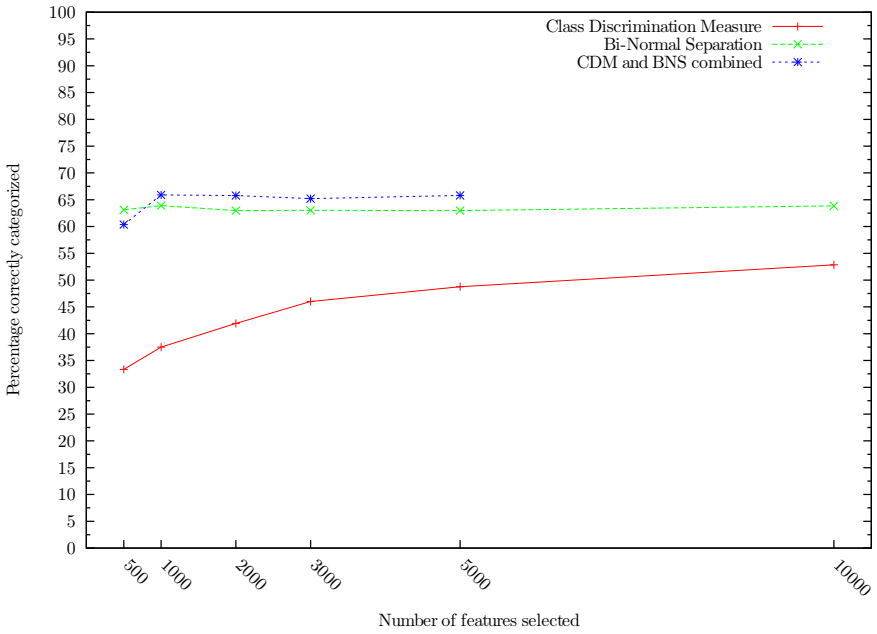

Figure 4.15: Percentage of documents correctly classified by a Naïve Bayes classifier using CDM, BNS,and the two combined on the 20 Newsgroups.

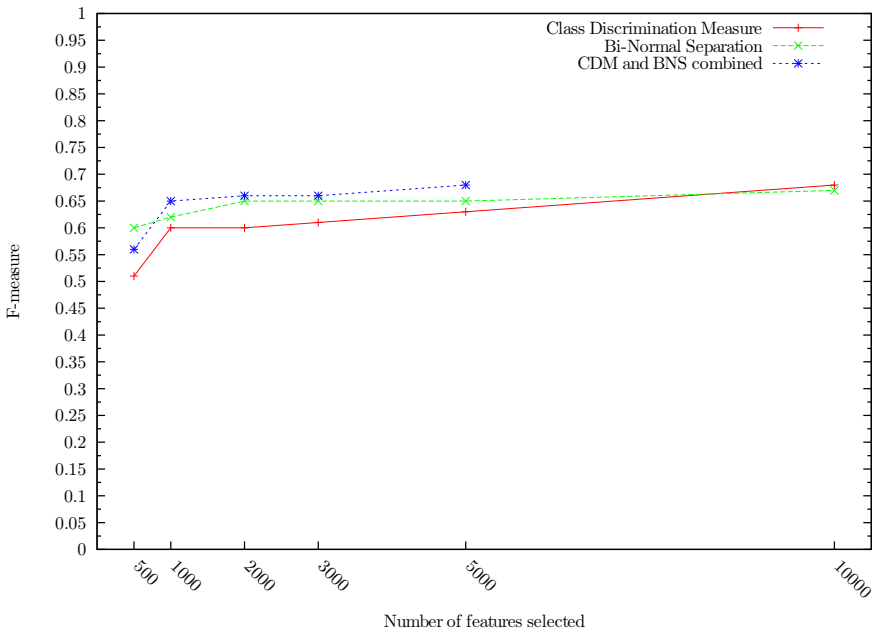

Figure 4.16: F-measure for a Naïve Bayes classifier using CDM, BNS, and the two combined at various numbers of features selected from the 20 Newsgroups.

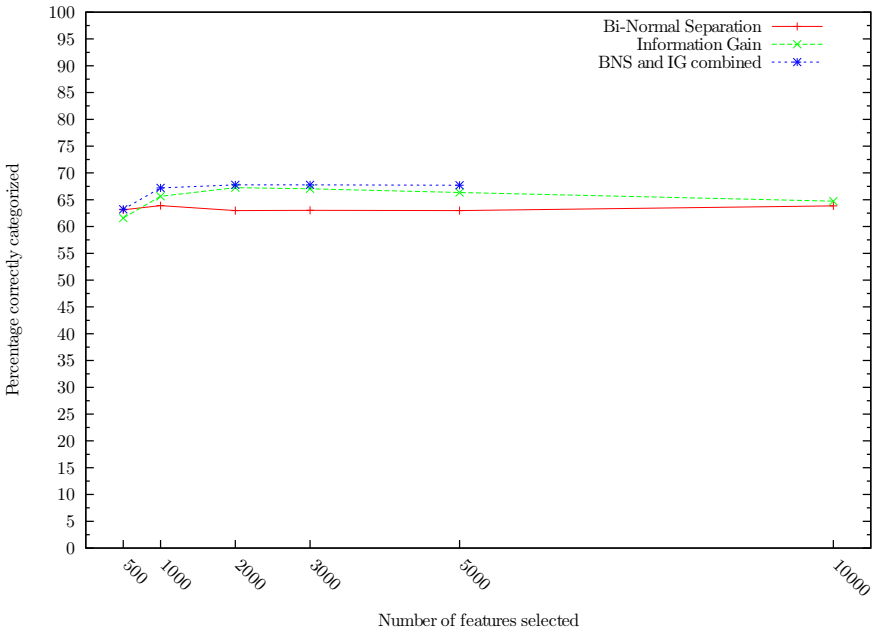

Figure 4.17: Percentage of documents correctly classified by a Naïve Bayes classifier using BNS, IG, and the two combined on the 20 Newsgroups.

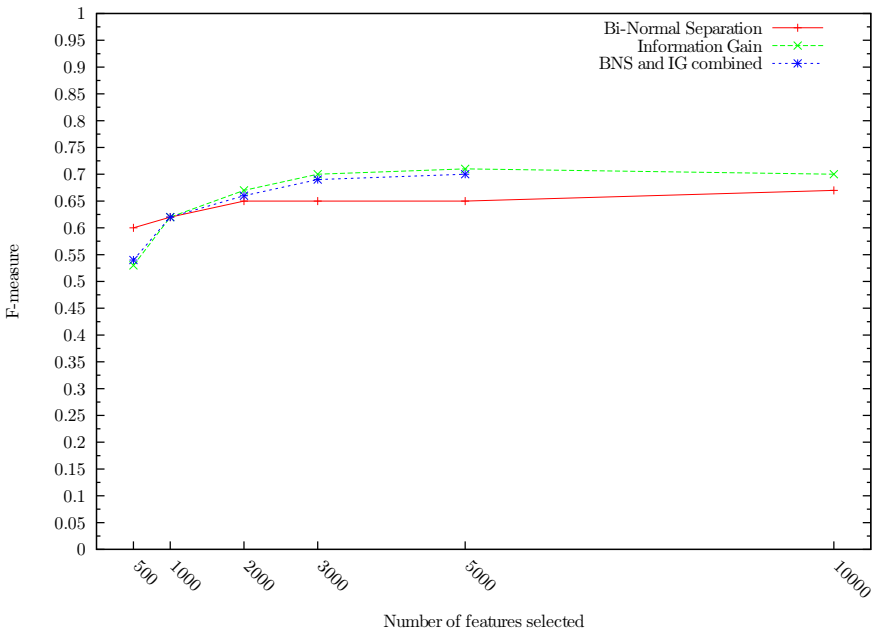

Figure 4.18: F-measure for a Naïve Bayes classifier using BNS, IG, and the two combined at various numbers of features selected from the 20 Newsgroups.

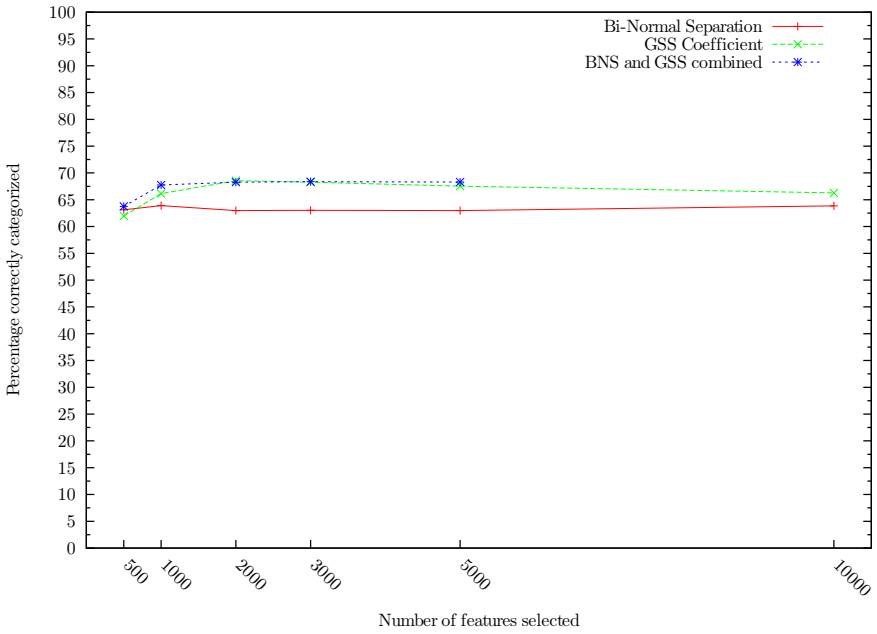

Figure 4.19: Percentage of documents correctly classified by a Naïve Bayes classifier using BNS, GSS, and the two combined on the 20 Newsgroups.

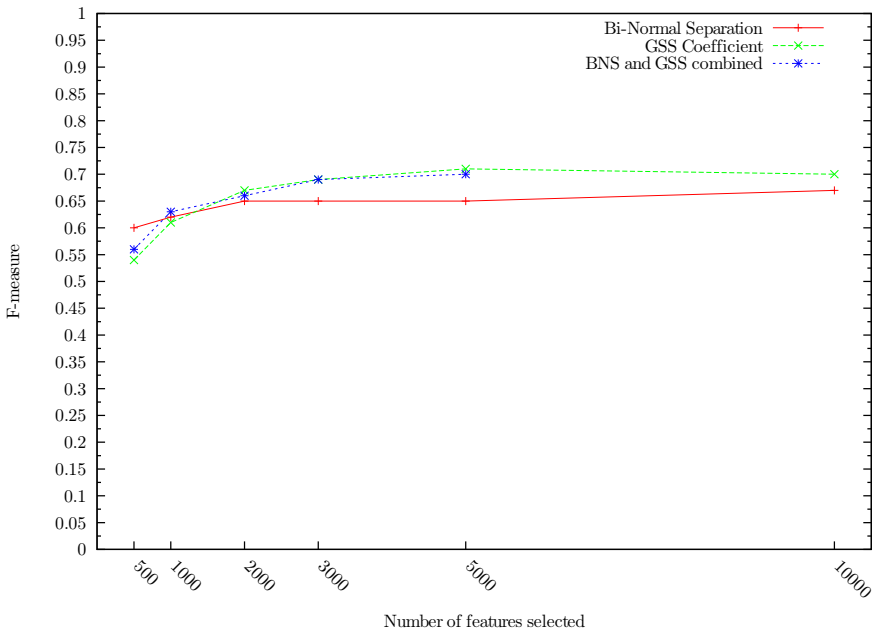

Figure 4.20: F-measure for a Naïve Bayes classifier using BNS, GSS, and the two combined at various numbers of features selected from the 20 Newsgroups.

Chapter 5

Conclusions and Outlook

We have evaluated 5 unsupervised and 11 supervised feature selection methods in addition to random selection on both a Naïve Bayes (NB) classifier and a Support Vector Machine (SVM) classifier, with various feature set sizes. We have also conducted a series of combination experiments where the joint effect of two or three feature selection methods were compared to each of the methods alone. In this section we make some concluding remarks about the results.

While most other feature selection research is presented using binary categorization tasks, our results are all from multi-class experiments with one common feature list for all classes. For this reason, our categorization results are not as good as others'. Also, our conclusions may not hold for binary experiments.

5.1 Conclusions

We start by concluding that SVM performs considerably better than NB, which confirms the findings of [Joa98]. The best feature selection methods (e.g. Chi Square and Information Gain) categorized more documents correctly on SVM with only 500 features than they did on NB on any of the feature set sizes we compared. Nevertheless, training an SVM classifier using 500 features, took longer time than building an NB classifier using 10000 features.

Furthermore, we conclude that feature selection is necessary, and that the choice of method is important. In fact, we were not able with our hardware to build a classifier with the full set of features of the 20 Newsgroups collection. Moreover, the results in Figures 4.1 through 4.8 clearly shows how very differently the metrics perform. While random feature selection obviously is not a good solution, the results of the DIA association factor shows worse performance than random indeed is possible.

The best results were achieved for both classifiers by CHI, GSS, IG, and MI. EOR was in the top group for NB, but not for SVM. The BNS metric was among the very best methods for the smallest feature sets, but did not produce the best results all over.

We have evaluated three unsupervised metrics previously not used for feature selection for text categorization: The Collection Frequency (CF), Collection Frequency Inverse Document Frequency (CFIDF), and a weirdness factor. Together with TFDF, CF and CFIDF produced almost as good results as the best

supervised methods, and can be recommended when pre-classified training data is not available. While the weirdness factor performed much better than random selection, its results are generally far from the other unsupervised methods like CF and CFIDF.

We found that the supervised method WF and the unsupervised method DF performed almost equally good on all feature set sizes and on both classifiers, suggesting that there is no need for the extra computations done in WF. Neither DF nor WF is recommended anyway, since as explained above we found both better supervised and unsupervised methods.

The worst results we found were those of DIA association factor and CPD in addition to random selection. These methods can hence not be recommended in the way that we used them. For the CPD metric, we discussed in Section 4.3.2 a possible way for it to be of practical use.

Our correlation matrix in Table 4.3 shows that very different methods can produce very good results. This means either that there are much redundancy in the information carried by a feature, or that there is still a way to go in feature selection. Our combination results mostly suggests the former, as most combinations did not cause much increase in performance. Some combinations were better however, so more research should be done in combining different methods. The BNS metric was a part of four of the five evaluated combinations that had a positive performance gain.

5.2 Further Work

In this section, we list some of the ideas we believe should be further investigated.

Additional corpora should be tested to see if our results hold for these as well. Text documents can contain very different features in different domains, and feature selection techniques may perform differently in other domains. We prepared the very different OHSUMED collection for experiments as described in Section 4.1.4, but did not have hardware resources enough to perform the experiments on this collection in addition to the 20 Newsgroups. The Reuters collections described in Sections 4.1.1 and 4.1.2 are also good candidates.

Additional feature selection methods should also be tested. Section 3.8 lists several methods we were not able to test due to time constraints.

Additional classifiers should be tested as well. The best feature selection method for one classifier may not be the best for another. The results of the EOR metric on NB and SVM serves as a good reason for evaluating feature selection on various induction schemes.

Other ways of globalizing the feature lists could be tested, including max, sum, average, round-robin (See [For04]) and other methods. It is important to remember that the same aggregation method might not be the best for various corpora. Moreover, a minimal test on very small feature sets might not be sufficient, as the the results may change when the feature sizes reach practically usable numbers. Large scale evaluation of globalization methods for every feature selection metric for various corpora is the only way to reliably find the best matches of globalization methods and feature selection methods. Obviously, this is a very time consuming exercise.

The weirdness factor should be further evaluated, as it shows promising results while selecting very different features than other methods. For instance, it should

be tested with CF instead of DF in the computation of feature values. Moreover, it should be tested with both upper and lower threshold. That is, instead of selecting the weirdest features only, one could filter out the weirdest features and select the ones following them. The motivation for this is that, as can be seen in our results, the weirdness factor performs poorly at the highest aggressivity levels, but shows very good results as the number of features increases. It could be that the features ranked highest were just *too* weird, and that performance might be increased without the weirdest ones.

Our experiments are all conducted without features that occur only in one document. Such 'rare feature cut-off' could be further investigated. While [For03] concludes that rare feature cutt-off might increase the recall values for the BNS metric, we have found no large-scale evaluations of the impact of such thresholds.

5.2.1 Combination Approaches

We conducted combination experiments as explained in Section 4.2.3 and evaluated in Section 4.3.3. Our results showed situations were the combined effort worked better than each of the individual methods, but also combinations with worse performance. Since BNS is a part of four of the five evaluated combinations that had a positive performance gain, it should be considered in future combination experiments. More feature selection combinations should be evaluated by the combination scheme we used. More than two or three methods should also be combined. The list of possible permutations is hence very long, but only large-scale testing could determine the benefits and drawbacks of this scheme.

Several other approaches to the combination of feature selection methods should be evaluated. An interesting combination scheme is the weighted sum of several normalized methods. For instance, if the feature values for each method were first normalized into a [0,1] interval, then several methods could be combined using $value = 0.2 \times IG + 0.5 \times CHI + 0.3 \times BNS$, where 0.2 etc. are example weights for how much an algorithm should count. This scheme share some similarities with the scheme presented in [RY02], albeit they use the max instead of a weighted average, and only from two feature selection metrics.

The concept of *sequential feature selection* should also be tested. By this we mean to use several methods in a sequence as filters, removing the lowest ranked features. The motivation for this, is that some methods seem to be very good at low aggressivity levels. For instance, the weirdness factor shows the best recall performance among all the methods we tested at a feature size of 10000, but considerably lower at higher aggressivity levels. Also, the CPD metric presented in [SH08] was there shown to outperform well-known methods like Information Gain, but with a feature set size of 61.2-77.3% of the available features. As such, it is possible that these methods are very good for determining the noisy features, while not so good at ranking the best features highest (see our discussion of the CPD metric in Section 4.3.2). Hence we propose to evaluate these metrics as thresholding metrics, followed by for instance Chi Square to do the actual ranking of the remaining features.

Sequential feature selection can be seen as an extension of the sequence of preprocessing steps that we and most other researchers perform. We already remove stop words, perform stemming, and remove features with low document frequency, all these are steps that reduce the feature space dimensionality. Sequential feature selection would in fact be a natural next step here. Removing

the least weird features suggested by the weirdness factor (in the bottom end of its ranked feature list) would serve as a natural extension to the stop word removal, as these features are so common in a general reference corpus that they may be of little categorization value. Which feature selection metrics that make sense to combine in sequence is only determined by actual large-scale experiments. The remaining features are probably best sorted by the same metrics that did well in comparison tests like the ones we present here, but this should also be tested.

Yet another possible combination scheme is the use of a voting system. Of m feature selection metrics, one can look past the actual feature weights and rather see the feature lists of the methods as votes. The feature occurring in the top position in *all* methods (if any) can be ranked on top in the combined ranking. Next, we include the second best ranking for all methods. If a feature now occurs in the top two positions for all metrics (and have not been added yet), it is added, and this continues until all features are ranked.

Alternatively, one could ease the voting system somewhat, demanding only n of the m metrics to suggest a feature before it is added to the combined list.

5.3 Contributions

We have compared a large number of feature selection methods on the same premises, in a scale that has not been done before. Where we compared 5 unsupervised and 11 supervised methods in addition to random selection, the second largest comparison test we have seen is that of [For03] where a total of 12 methods including random selection were tested. All methods we tested were evaluated on two important classifiers.

Our experiments also include metrics that have not been used for feature selection before, including the CF, CFIDF and weirdness factor. The CF and CFIDF results are among the best for unsupervised methods, and are close to the best supervised methods.

While most other research in feature selection presents binary categorization tasks using one feature list for each class, we present here multi-class result. That is, our results are based training classifiers using one global feature list for all classes. As we conducted our experiments on the 20-Newsgroups corpus with 20 classes, the performance reported naturally is not as high as for binary tasks with local feature lists. However, our results are important for anyone in need of multi-class classification using global lists, and it shows which feature selection methods that are prone to error when globalizing the feature lists.

Our experiments were done on the entire 20-Newsgroups corpus. While there exists several larger corpora, most researchers use small subsets of these.

We used 10-fold cross-validation averaged over 5 or 10 runs (a total of 50 to 100 passes over the dataset for each feature set size), while most other researchers have been using one designated split. Hence, our results should be very reliable, and reproducible.

We have conducted a range of combination experiments, and the best results we achieved for NB was by combining the CHI and GSS metrics.

Bibliography

[AGT99] Khurshid Ahmad, Lee Gillam, and Lena Tostevin. Weirdness indexing for logical document extrapolation and retrieval WILDER. In *TREC*, 1999.

[BFH+08] Remco R. Bouckaert, Eibe Frank, Mark Hall, Richard Kirkby, Peter Reutemann, Alex Seewald, and David Scuse. Weka manual for version 3-6-0. The University of Waikato, Hamilton, New Zealand, December 2008.

[BMGMF08] Janez Brank, Dunja Mladenić, Marko Grobelnik, and Nataša Milić-Frayling. Feature selection for the classification of large document collections. *Journal of Universal Computer Science*, 14(10):1562–1596, May 2008.

[BYRN99] Ricardo A. Baeza-Yates and Berthier A. Ribeiro-Neto. *Modern Information Retrieval*. ACM Press / Addison-Wesley, 1999.

[CHTQ09] Jingnian Chen, Houkuan Huang, Shengfeng Tian, and Youli Qu. Feature selection for text classification with naïve bayes. *Expert Systems with Applications*, 36(3):5432–5435, 2009. doi:10.1016/j.eswa.2008.06.054.

[Cim06] Philipp Cimiano. *Ontology Learning and Population from Text: Algorithms, Evaluation and Applications*. Springer-Verlag, New York, NY, USA, 2006.

[CMS01] Maria F. Caropreso, Stan Matwin, and Fabrizio Sebastiani. A learner-independent evaluation of the usefulness of statistical phrases for automated text categorization. In Amita G. Chin, editor, *Text Databases and Document Management: Theory and Practice*, pages 78–102. Idea Group Publishing, Hershey, US, 2001.

[CS08] Jihong Cai and Fei Song. Maximum entropy modeling with feature selection for text categorization. In Li et al. [LLM+08], pages 549–554.

[DC00] Susan T. Dumais and Hao Chen. Hierarchical classification of Web content. In Nicholas J. Belkin, Peter Ingwersen, and Mun-Kew Leong, editors, *Proceedings of SIGIR-00, 23rd ACM International Conference on Research and Development in Information Retrieval*, pages 256–263, Athens, GR, 2000. ACM Press, New York, US.

[DDH+07] Anirban Dasgupta, Petros Drineas, Boulos Harb, Vanja Josifovski, and Michael W. Mahoney. Feature selection methods for text classification. In Pavel Berkhin, Rich Caruana, and Xindong Wu, editors, *KDD*, pages 230–239. ACM, 2007.

[EM07] Susana Eyheramendy and David Madigan. A bayesian feature selection score based on naïve bayes models. In Liu and Motoda [LM07a], pages 277–294.

[FHK+91] Norbert Fuhr, Stephan Hartmann, Gerhard Knorz, Gerhard Lustig, Michael Schwantner, and Konstadinos Tzeras. AIR/X - a rule-based multistage indexing system for large subject fields. In *Proceedings of the RIAO'91, Barcelona, Spain, April 2-5, 1991*, pages 606–623, 1991.

[For02] George Forman. Choose your words carefully: An empirical study of feature selection metrics for text classification. In Tapio Elomaa, Heikki Mannila, and Hannu Toivonen, editors, *PKDD*, volume 2431 of *Lecture Notes in Computer Science*, pages 150–162. Springer, 2002.

[For03] George Forman. An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.*, 3:1289–1305, 2003.

[For04] George Forman. A pitfall and solution in multi-class feature selection for text classification. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 38, New York, NY, USA, 2004. ACM.

[For07] George Forman. Feature selection for text classification. In Liu and Motoda [LM07a], pages 257–276.

[FS07] Ronen Feldman and James Sanger. *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, New York, NY, USA, 2007.

[GLM07] Alexander Genkin, David D. Lewis, and David Madigan. Large-scale bayesian logistic regression for text categorization. *Technometrics*, 49:291–304(14), August 2007.

[GSS00] Luigi Galavotti, Fabrizio Sebastiani, and Maria Simi. Experiments on the use of feature selection and negative evidence in automated text categorization. In José Luis Borbinha and Thomas Baker, editors, *ECDL*, volume 1923 of *Lecture Notes in Computer Science*, pages 59–68. Springer, 2000.

[HBLH94] William Hersh, Chris Buckley, T. J. Leone, and David Hickam. Ohsumed: an interactive retrieval evaluation and new large test collection for research. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 192–201. Springer-Verlag New York, Inc., 1994.

[HK00] Eui-Hong Han and George Karypis. Centroid-based document classification: Analysis and experimental results. In Djamel A. Zighed, Henryk Jan Komorowski, and Jan M. Zytkow, editors, *PKDD*, volume 1910 of *Lecture Notes in Computer Science*, pages 424–431. Springer, 2000.

[Joa97] T. Joachims. A probabilistic analysis of the rocchio algorithm with tfidf for text categorization. In *Proceedings of the 14th International Conference on Machine Learning*, 1997.

[Joa98] Thorsten Joachims. Text categorization with support vector machines: learning with many relevant features. In Claire Nedellec and Cline Rouveirol, editors, *Proceedings of ECML-98, 10th European Conference on Machine Learning*, pages 137–142, Heidelberg et al., 1998. Springer.

[KS97] Daphne Koller and Mehran Sahami. Hierarchically classifying documents using very few words. In *Proceedings of the 14th International Conference on Machine Learning (ICML'97)*, pages 170–178, 1997.

[KYMW97] Toshiki Kindo, Hideyuki Yoshida, Tetsuro Morimoto, and Taisuke Watanabe. Adaptive personal information filtering system that organizes personal profiles automatically. In *IJCAI (1)*, pages 716–721, 1997.

[Lan95] Ken Lang. Newsweeder: Learning to filter netnews. In *Proceedings of the Twelfth International Conference on Machine Learning*, pages 331–339, 1995.

[Lew92] David D. Lewis. An evaluation of phrasal and clustered representations on a text categorization task. In *SIGIR '92: Proceedings of the 15th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 37–50, New York, NY, USA, 1992. ACM Press.

[LJ98] Yonghong Li and Anil K. Jain. Classification of text documents. *The Computer Journal*, 41(8):537–546, 1998.

[LLM+08] Hang Li, Ting Liu, Wei-Ying Ma, Tetsuya Sakai, Kam-Fai Wong, and Guodong Zhou, editors. *Information Retrieval Technology, 4th Asia Infomation Retrieval Symposium, AIRS 2008, Harbin, China, January 15-18, 2008, Revised Selected Papers*, volume 4993 of *Lecture Notes in Computer Science*. Springer, 2008.

[LM07a] Huan Liu and Hiroshi Motoda, editors. *Computational Methods of Feature Selection*. Data Mining and Knowledge Discovery series. Chapman & Hall/CRC, Boca Raton, FL, October 2007.

[LM07b] Huan Liu and Hiroshi Motoda. Less is more. In *Computational Methods of Feature Selection* [LM07a], pages 3–18.

[LR94] David D. Lewis and Marc Ringuette. A comparison of two learning algorithms for text categorization. In *Proceedings of SDAIR'94: 3rd Annual Symposium on Document Analysis and Information Retrieval*, pages 81–93, Las Vegas, NV, USA, 1994.

[LTSL07] Man Lan, Chew Lim Tan, Jian Su, and Hwee-Boon Low. Text representations for text categorization: A case study in biomedical domain. In *IJCNN*, pages 2557–2562. IEEE, 2007.

[LYRL04] David D. Lewis, Yiming Yang, Tony G. Rose, and Fan Li. RCV1: A new benchmark collection for text categorization research. *Journal of Machine Learning Research*, 5:361–397, 2004.

[MK07] Masoud Makrehchi and Mohammed S. Kamel. Aggressive feature selection by feature ranking. In Liu and Motoda [LM07a], pages 313–333.

[Mla98] Dunja Mladenić. Feature subset selection in text-learning. In *Proceedings of the 10th European Conference on Machine Learning (ECML'98)*, pages 95–100, London, England, UK, 1998. Springer-Verlag.

[MRS08] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. Cambridge University Press, Cambridge, UK, 2008.

[NGL97] Hwee Tou Ng, Wei Boon Goh, and Kok Leong Low. Feature selection, perceptron learning, and a usability case study for text categorization. In Nicholas J. Belkin, A. Desai Narasimhalu, and Peter Willett, editors, *Proceedings of SIGIR-97, 20th ACM International Conference on Research and Development in Information Retrieval*, pages 67–73, Philadelphia, US, 1997. ACM Press, New York, US.

[Por80] Martin F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, 1980.

[Qui86] John Ross Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[RS99] Miguel E. Ruiz and Padmini Srinivasan. Hierarchical neural networks for text categorization (poster abstract). In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 281–282, New York, NY, USA, 1999. ACM.

[RY02] Monica Rogati and Yiming Yang. High-performing feature selection for text classification. In *CIKM*, pages 659–661. ACM, 2002.

[Seb02] Fabrizio Sebastiani. Machine learning in automated text categorization. *ACM Computing Surveys*, 34(1):1–47, 2002.

[SH08] Mondelle Simeon and Robert J. Hilderman. Categorical proportional difference: A feature selection method for text categorization. In John F. Roddick, Jiuyong Li, Peter Christen, and Paul J. Kennedy, editors, *AusDM*, volume 87 of *CRPIT*, pages 201–208. Australian Computer Society, 2008.

[Sha95] William M. Shaw, Jr. Term-relevance computations and perfect retrieval performance. *Information Processing and Management: an International Journal*, 31(4):491–498, 1995.

[vR79] C. J. van Rijsbergen. *Information retrieval*. Butterworths, London, 2 edition, 1979.

[Wit08] Hans Friedrich Witschel. Global term weights in distributed environments. *Information Processing and Management*, 44(3):1049–1061, 2008.

[XWLJ08] Yan Xu, Bin Wang, Jintao Li, and Hongfang Jing. An extended document frequency metric for feature selection in text categorization. In Li et al. [LLM+08], pages 71–82.

[YH08] Shuang-Hong Yang and Bao-Gang Hu. Efficient feature selection in the presence of outliers and noises. In Li et al. [LLM+08], pages 184–191.

[YP97] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *ICML '97: Proceedings of the Fourteenth International Conference on Machine Learning*, pages 412–420, San Francisco, CA, USA, 1997. Morgan Kaufmann Publishers Inc.

[ZH07] Hui Zou and Trevor Hastie. Model building and feature selection with genomic data. In Liu and Motoda [LM07a], pages 393–411.

[ZZH04] Q. Zhou, M. ZHao, and M. Hu. Study on feature selection in chinese text categorization. *Journal of Chinese Information Processing*, 18:17–23, 2004.

Appendix A

Corpora

This appendix will include information related to the text collections used.

A.1 20-Newsgroups

The following text is an example of a post from the 20-Newsgroups collection. This message was posted to the newsgroup (or category) `comp.sys.mac.hardware`.

```
From: jcav@ellis.uchicago.edu (JohnC)
Subject: how do you like the Apple Color OneScanner?

We're all set to buy one of these for the office, to use for scanning in
color photographs and for optical character recognition.  We've played with
the original grayscale OneScanner and were very pleased.  Is the color model
comparable in quality?

Also, what brand of OCR software would you recommend?  We're leaning toward
Caere OmniPage.  Any better ideas?  Thanks.

--
John Cavallino                  |  EMail: jcav@midway.uchicago.edu
University of Chicago Hospitals |         John_Cavallino@uchfm.bsd.uchicago.edu
Office of Facilities Management | USMail: 5841 S. Maryland Ave, MC 0953
BO f++ w c+ g++ k+ s++ e h- p   |         Chicago, IL  60637
```

A.2 OHSUMED

The following text is an example of a reference (MEDLINE ID 91076390) from the OHSUMED collection. We extracted only the heading and the abstract fields.

```
Ethics consultation: skills, roles, and training [see comments]

A clinical ethics consultant gathers information firsthand at the
patient's bedside. The consultant's special clinical skills include the
```

ability to identify and analyze ethical problems; use reasonable clinical judgment; communicate effectively; negotiate and facilitate negotiations; and teach others how to construct their own ethical frameworks for medical decision making. Appropriate roles for the consultant include those of professional colleague, negotiator, patient and physician advocate, case manager, and educator. The training necessary for an ethics consultant includes substantial patient care experience, instruction in health care law and moral reasoning, and preparation in medical humanism. We favor a clinical model for ethics consultation. When urgent care is needed, other consultants promptly see the patient; the clinical ethics consultant can be expected to do the same.

Appendix B

Experiments

This appendix will list the detailed results of the experiments, and some graphs from comparison experiments.

B.1 Result Tables

In this section, the detailed results from the experiments are listed. Tables B.1 through B.8 contain standard deviations, that measure the variability of the results. Standard deviations are calculated by first calculating the difference of each run's result from the mean (average) of all runs. Next each difference is squared, and these values are then averaged over all runs. Finally, we take the square root of this average value, which gives the standard deviation.

Table B.1: Percentage correctly categorized documents using a Naïve Bayes classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 10 runs.

Feature selectiom method and feature set size	Percent correct		
RANDOM-500	7.42	±	0.33
RANDOM-1000	7.96	±	0.43
RANDOM-2000	13.33	±	0.61
RANDOM-3000	16.08	±	0.72
RANDOM-5000	21.68	±	0.90
RANDOM-10000	28.09	±	0.93
COLLECTION-FREQUENCY-500	54.21	±	1.14
COLLECTION-FREQUENCY-1000	60.64	±	1.15
COLLECTION-FREQUENCY-2000	64.15	±	1.04
COLLECTION-FREQUENCY-3000	64.08	±	1.01
COLLECTION-FREQUENCY-5000	64.04	±	1.02
COLLECTION-FREQUENCY-10000	64.05	±	1.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	54.18	±	1.11
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	61.18	±	1.07
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	64.29	±	1.07
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-3000	64.27	±	1.02
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-5000	64.18	±	0.93
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-10000	65.09	±	1.08
DOCUMENT-FREQUENCY-500	49.01	±	1.09
DOCUMENT-FREQUENCY-1000	57.82	±	1.06
DOCUMENT-FREQUENCY-2000	62.31	±	1.02
DOCUMENT-FREQUENCY-3000	62.97	±	1.00
DOCUMENT-FREQUENCY-5000	63.37	±	1.03
DOCUMENT-FREQUENCY-10000	64.29	±	1.10
TERM-FREQUENCY-DOCUMENT-FREQUENCY-500	53.09	±	1.10
TERM-FREQUENCY-DOCUMENT-FREQUENCY-1000	61.30	±	1.02
TERM-FREQUENCY-DOCUMENT-FREQUENCY-2000	64.88	±	0.96

Continued on next page

Table B.1 – continued from previous page

Feature selectiom method and feature set size	Percent correct		
TERM-FREQUENCY-DOCUMENT-FREQUENCY-3000	65.54	±	1.08
TERM-FREQUENCY-DOCUMENT-FREQUENCY-5000	65.63	±	0.98
TERM-FREQUENCY-DOCUMENT-FREQUENCY-10000	65.37	±	1.04
WEIRDNESS-500	30.50	±	0.99
WEIRDNESS-1000	34.70	±	0.93
WEIRDNESS-2000	42.35	±	1.02
WEIRDNESS-3000	45.02	±	1.12
WEIRDNESS-5000	49.34	±	1.10
WEIRDNESS-10000	61.88	±	1.10
WORD-FREQUENCY-500	48.84	±	1.11
WORD-FREQUENCY-1000	57.98	±	1.08
WORD-FREQUENCY-2000	62.34	±	1.01
WORD-FREQUENCY-3000	63.12	±	0.99
WORD-FREQUENCY-5000	63.43	±	1.05
WORD-FREQUENCY-10000	64.33	±	1.07
INFORMATION-GAIN-500	61.59	±	1.01
INFORMATION-GAIN-1000	65.64	±	1.08
INFORMATION-GAIN-2000	67.25	±	0.98
INFORMATION-GAIN-3000	67.03	±	1.02
INFORMATION-GAIN-5000	66.35	±	1.01
INFORMATION-GAIN-10000	64.73	±	1.12
MUTUAL-INFORMATION-500	61.69	±	1.05
MUTUAL-INFORMATION-1000	66.30	±	0.99
MUTUAL-INFORMATION-2000	67.19	±	1.01
MUTUAL-INFORMATION-3000	67.32	±	1.00
MUTUAL-INFORMATION-5000	66.86	±	1.03
MUTUAL-INFORMATION-10000	65.10	±	1.08
ODDS-RATIO-500	61.81	±	1.03
ODDS-RATIO-1000	66.52	±	1.07
ODDS-RATIO-2000	68.24	±	1.03
ODDS-RATIO-3000	67.85	±	0.98
ODDS-RATIO-5000	67.36	±	1.05
ODDS-RATIO-10000	59.86	±	0.96
CLASS-DISCRIMINATION-MEASURE-500	33.33	±	0.84
CLASS-DISCRIMINATION-MEASURE-1000	37.51	±	0.87
CLASS-DISCRIMINATION-MEASURE-2000	41.93	±	0.98
CLASS-DISCRIMINATION-MEASURE-3000	46.00	±	0.98
CLASS-DISCRIMINATION-MEASURE-5000	48.76	±	1.03
CLASS-DISCRIMINATION-MEASURE-10000	52.85	±	1.15
CHI-SQUARE-500	64.21	±	1.01
CHI-SQUARE-1000	67.06	±	0.95
CHI-SQUARE-2000	68.78	±	1.02
CHI-SQUARE-3000	**69.19**	±	0.96
CHI-SQUARE-5000	68.74	±	1.09
CHI-SQUARE-10000	67.06	±	1.04
NGL-COEFFICIENT-500	44.67	±	1.11
NGL-COEFFICIENT-1000	46.79	±	1.05
NGL-COEFFICIENT-2000	45.29	±	1.16
NGL-COEFFICIENT-3000	45.39	±	1.00
NGL-COEFFICIENT-5000	47.06	±	1.14
NGL-COEFFICIENT-10000	48.68	±	1.06
GSS-COEFFICIENT-500	61.98	±	1.16
GSS-COEFFICIENT-1000	66.14	±	0.99
GSS-COEFFICIENT-2000	68.55	±	0.90
GSS-COEFFICIENT-3000	68.27	±	0.96
GSS-COEFFICIENT-5000	67.53	±	1.07
GSS-COEFFICIENT-10000	66.28	±	1.14
BI-NORMAL-SEPARATION-500	63.11	±	0.92
BI-NORMAL-SEPARATION-1000	63.90	±	0.85
BI-NORMAL-SEPARATION-2000	62.98	±	0.94
BI-NORMAL-SEPARATION-3000	63.02	±	0.97
BI-NORMAL-SEPARATION-5000	62.98	±	0.96
BI-NORMAL-SEPARATION-10000	63.85	±	0.88
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	8.83	±	0.40
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	9.60	±	0.50
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	10.54	±	0.64
CATEGORICAL-PROPORTIONAL-DIFFERENCE-3000	11.99	±	0.61
CATEGORICAL-PROPORTIONAL-DIFFERENCE-5000	18.48	±	0.60
CATEGORICAL-PROPORTIONAL-DIFFERENCE-10000	22.13	±	0.89
DIA-ASSOCIATION-FACTOR-500	5.42	±	0.08
DIA-ASSOCIATION-FACTOR-1000	6.11	±	0.19

Continued on next page

Table B.1 – continued from previous page

Feature selectiom method and feature set size	Percent correct		
DIA-ASSOCIATION-FACTOR-2000	6.75	±	0.22
DIA-ASSOCIATION-FACTOR-3000	7.50	±	0.23
DIA-ASSOCIATION-FACTOR-5000	7.43	±	0.40
DIA-ASSOCIATION-FACTOR-10000	11.44	±	0.49

Table B.2: Macroaverage precision of a Naïve Bayes classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 10 runs.

Feature selectiom method and feature set size	Macroavg. precision		
RANDOM-500	0.14	±	0.06
RANDOM-1000	0.26	±	0.20
RANDOM-2000	0.39	±	0.12
RANDOM-3000	0.34	±	0.10
RANDOM-5000	0.21	±	0.05
RANDOM-10000	0.27	±	0.06
COLLECTION-FREQUENCY-500	0.51	±	0.04
COLLECTION-FREQUENCY-1000	0.57	±	0.05
COLLECTION-FREQUENCY-2000	0.65	±	0.05
COLLECTION-FREQUENCY-3000	0.68	±	0.04
COLLECTION-FREQUENCY-5000	0.71	±	0.05
COLLECTION-FREQUENCY-10000	0.71	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	0.51	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	0.59	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	0.64	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-3000	0.68	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-5000	0.72	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-10000	0.69	±	0.05
DOCUMENT-FREQUENCY-500	0.41	±	0.05
DOCUMENT-FREQUENCY-1000	0.56	±	0.05
DOCUMENT-FREQUENCY-2000	0.65	±	0.05
DOCUMENT-FREQUENCY-3000	0.68	±	0.05
DOCUMENT-FREQUENCY-5000	0.72	±	0.05
DOCUMENT-FREQUENCY-10000	0.71	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-500	0.51	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-1000	0.58	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-2000	0.64	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-3000	0.67	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-5000	0.69	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-10000	0.70	±	0.05
WEIRDNESS-500	0.37	±	0.06
WEIRDNESS-1000	0.39	±	0.06
WEIRDNESS-2000	0.56	±	0.06
WEIRDNESS-3000	0.56	±	0.06
WEIRDNESS-5000	0.48	±	0.07
WEIRDNESS-10000	0.55	±	0.04
WORD-FREQUENCY-500	0.40	±	0.05
WORD-FREQUENCY-1000	0.54	±	0.05
WORD-FREQUENCY-2000	0.64	±	0.05
WORD-FREQUENCY-3000	0.67	±	0.05
WORD-FREQUENCY-5000	0.71	±	0.05
WORD-FREQUENCY-10000	0.71	±	0.05
INFORMATION-GAIN-500	0.58	±	0.05
INFORMATION-GAIN-1000	0.64	±	0.05
INFORMATION-GAIN-2000	0.72	±	0.05
INFORMATION-GAIN-3000	0.73	±	0.05
INFORMATION-GAIN-5000	0.74	±	0.05
INFORMATION-GAIN-10000	0.71	±	0.04
MUTUAL-INFORMATION-500	0.58	±	0.05
MUTUAL-INFORMATION-1000	0.63	±	0.05
MUTUAL-INFORMATION-2000	0.69	±	0.05
MUTUAL-INFORMATION-3000	0.73	±	0.04
MUTUAL-INFORMATION-5000	0.73	±	0.05
MUTUAL-INFORMATION-10000	0.71	±	0.04
ODDS-RATIO-500	0.57	±	0.04
ODDS-RATIO-1000	0.63	±	0.05
ODDS-RATIO-2000	0.70	±	0.05
ODDS-RATIO-3000	0.72	±	0.05
ODDS-RATIO-5000	0.72	±	0.05

Continued on next page

Table B.2 – continued from previous page

Feature selectiom method and feature set size	Macroavg. precision		
ODDS-RATIO-10000	0.67	±	0.05
CLASS-DISCRIMINATION-MEASURE-500	0.96	±	0.03
CLASS-DISCRIMINATION-MEASURE-1000	0.90	±	0.04
CLASS-DISCRIMINATION-MEASURE-2000	0.89	±	0.05
CLASS-DISCRIMINATION-MEASURE-3000	0.89	±	0.05
CLASS-DISCRIMINATION-MEASURE-5000	0.87	±	0.05
CLASS-DISCRIMINATION-MEASURE-10000	0.80	±	0.05
CHI-SQUARE-500	0.66	±	0.05
CHI-SQUARE-1000	0.68	±	0.05
CHI-SQUARE-2000	0.69	±	0.05
CHI-SQUARE-3000	0.71	±	0.05
CHI-SQUARE-5000	0.75	±	0.05
CHI-SQUARE-10000	0.72	±	0.05
NGL-COEFFICIENT-500	0.10	±	0.03
NGL-COEFFICIENT-1000	0.10	±	0.03
NGL-COEFFICIENT-2000	0.10	±	0.02
NGL-COEFFICIENT-3000	0.10	±	0.02
NGL-COEFFICIENT-5000	0.10	±	0.02
NGL-COEFFICIENT-10000	0.10	±	0.02
GSS-COEFFICIENT-500	0.60	±	0.05
GSS-COEFFICIENT-1000	0.65	±	0.05
GSS-COEFFICIENT-2000	0.68	±	0.04
GSS-COEFFICIENT-3000	0.70	±	0.05
GSS-COEFFICIENT-5000	0.71	±	0.05
GSS-COEFFICIENT-10000	0.70	±	0.05
BI-NORMAL-SEPARATION-500	0.67	±	0.05
BI-NORMAL-SEPARATION-1000	0.69	±	0.05
BI-NORMAL-SEPARATION-2000	0.71	±	0.05
BI-NORMAL-SEPARATION-3000	0.71	±	0.04
BI-NORMAL-SEPARATION-5000	0.72	±	0.05
BI-NORMAL-SEPARATION-10000	0.72	±	0.05
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	0.77	±	0.42
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	0.77	±	0.42
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	0.77	±	0.42
CATEGORICAL-PROPORTIONAL-DIFFERENCE-3000	0.87	±	0.34
CATEGORICAL-PROPORTIONAL-DIFFERENCE-5000	**1.00**	±	0.00
CATEGORICAL-PROPORTIONAL-DIFFERENCE-10000	**1.00**	±	0.00
DIA-ASSOCIATION-FACTOR-500	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-1000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-2000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-3000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-5000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-10000	0.00	±	0.00

Table B.3: Macroaverage recall of a Naïve Bayes classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 10 runs.

Feature selectiom method and feature set size	Macroavg. recall		
RANDOM-500	0.05	±	0.02
RANDOM-1000	0.02	±	0.01
RANDOM-2000	0.08	±	0.03
RANDOM-3000	0.08	±	0.03
RANDOM-5000	0.19	±	0.04
RANDOM-10000	0.19	±	0.05
COLLECTION-FREQUENCY-500	0.46	±	0.06
COLLECTION-FREQUENCY-1000	0.53	±	0.05
COLLECTION-FREQUENCY-2000	0.62	±	0.05
COLLECTION-FREQUENCY-3000	0.66	±	0.05
COLLECTION-FREQUENCY-5000	0.67	±	0.05
COLLECTION-FREQUENCY-10000	0.70	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	0.48	±	0.06
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	0.54	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	0.62	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-3000	0.65	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-5000	0.66	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-10000	0.70	±	0.05
DOCUMENT-FREQUENCY-500	0.40	±	0.06
DOCUMENT-FREQUENCY-1000	0.50	±	0.05

Continued on next page

Table B.3 – continued from previous page

Feature selectiom method and feature set size	Macroavg. recall		
DOCUMENT-FREQUENCY-2000	0.62	±	0.06
DOCUMENT-FREQUENCY-3000	0.65	±	0.05
DOCUMENT-FREQUENCY-5000	0.67	±	0.05
DOCUMENT-FREQUENCY-10000	0.69	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-500	0.46	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-1000	0.54	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-2000	0.63	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-3000	0.66	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-5000	0.69	±	0.05
TERM-FREQUENCY-DOCUMENT-FREQUENCY-10000	0.70	±	0.05
WEIRDNESS-500	0.27	±	0.05
WEIRDNESS-1000	0.29	±	0.05
WEIRDNESS-2000	0.45	±	0.06
WEIRDNESS-3000	0.44	±	0.06
WEIRDNESS-5000	0.62	±	0.06
WEIRDNESS-10000	**0.72**	±	0.05
WORD-FREQUENCY-500	0.39	±	0.06
WORD-FREQUENCY-1000	0.50	±	0.05
WORD-FREQUENCY-2000	0.61	±	0.06
WORD-FREQUENCY-3000	0.64	±	0.06
WORD-FREQUENCY-5000	0.67	±	0.05
WORD-FREQUENCY-10000	0.70	±	0.05
INFORMATION-GAIN-500	0.50	±	0.05
INFORMATION-GAIN-1000	0.60	±	0.05
INFORMATION-GAIN-2000	0.64	±	0.06
INFORMATION-GAIN-3000	0.66	±	0.05
INFORMATION-GAIN-5000	0.69	±	0.05
INFORMATION-GAIN-10000	0.70	±	0.05
MUTUAL-INFORMATION-500	0.50	±	0.05
MUTUAL-INFORMATION-1000	0.57	±	0.05
MUTUAL-INFORMATION-2000	0.64	±	0.05
MUTUAL-INFORMATION-3000	0.66	±	0.05
MUTUAL-INFORMATION-5000	0.70	±	0.05
MUTUAL-INFORMATION-10000	0.70	±	0.05
ODDS-RATIO-500	0.50	±	0.05
ODDS-RATIO-1000	0.60	±	0.05
ODDS-RATIO-2000	0.65	±	0.05
ODDS-RATIO-3000	0.67	±	0.05
ODDS-RATIO-5000	0.70	±	0.05
ODDS-RATIO-10000	0.63	±	0.05
CLASS-DISCRIMINATION-MEASURE-500	0.35	±	0.05
CLASS-DISCRIMINATION-MEASURE-1000	0.46	±	0.06
CLASS-DISCRIMINATION-MEASURE-2000	0.46	±	0.06
CLASS-DISCRIMINATION-MEASURE-3000	0.47	±	0.06
CLASS-DISCRIMINATION-MEASURE-5000	0.50	±	0.06
CLASS-DISCRIMINATION-MEASURE-10000	0.59	±	0.06
CHI-SQUARE-500	0.56	±	0.05
CHI-SQUARE-1000	0.60	±	0.05
CHI-SQUARE-2000	0.65	±	0.05
CHI-SQUARE-3000	0.69	±	0.05
CHI-SQUARE-5000	0.69	±	0.05
CHI-SQUARE-10000	0.71	±	0.05
NGL-COEFFICIENT-500	0.11	±	0.03
NGL-COEFFICIENT-1000	0.16	±	0.07
NGL-COEFFICIENT-2000	0.21	±	0.05
NGL-COEFFICIENT-3000	0.27	±	0.09
NGL-COEFFICIENT-5000	0.29	±	0.11
NGL-COEFFICIENT-10000	0.32	±	0.11
GSS-COEFFICIENT-500	0.50	±	0.05
GSS-COEFFICIENT-1000	0.58	±	0.05
GSS-COEFFICIENT-2000	0.65	±	0.05
GSS-COEFFICIENT-3000	0.69	±	0.05
GSS-COEFFICIENT-5000	**0.71**	±	0.05
GSS-COEFFICIENT-10000	0.70	±	0.05
BI-NORMAL-SEPARATION-500	0.54	±	0.05
BI-NORMAL-SEPARATION-1000	0.57	±	0.05
BI-NORMAL-SEPARATION-2000	0.60	±	0.05
BI-NORMAL-SEPARATION-3000	0.60	±	0.05
BI-NORMAL-SEPARATION-5000	0.60	±	0.05
BI-NORMAL-SEPARATION-10000	0.63	±	0.05
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	0.02	±	0.01

Continued on next page

Table B.3 – continued from previous page

Feature selectiom method and feature set size	Macroavg. recall		
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	0.02	±	0.01
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	0.02	±	0.01
CATEGORICAL-PROPORTIONAL-DIFFERENCE-3000	0.03	±	0.02
CATEGORICAL-PROPORTIONAL-DIFFERENCE-5000	0.08	±	0.03
CATEGORICAL-PROPORTIONAL-DIFFERENCE-10000	0.19	±	0.04
DIA-ASSOCIATION-FACTOR-500	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-1000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-2000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-3000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-5000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-10000	0.00	±	0.00

Table B.4: Macroaverage F_1-measure of a Naïve Bayes classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 10 runs.

Feature selectiom method and feature set size	Macroavg. F_1		
RANDOM-500	0.08	±	0.03
RANDOM-1000	0.03	±	0.02
RANDOM-2000	0.12	±	0.04
RANDOM-3000	0.13	±	0.04
RANDOM-5000	0.20	±	0.04
RANDOM-10000	0.22	±	0.05
COLLECTION-FREQUENCY-500	0.48	±	0.05
COLLECTION-FREQUENCY-1000	0.55	±	0.04
COLLECTION-FREQUENCY-2000	0.63	±	0.04
COLLECTION-FREQUENCY-3000	0.67	±	0.04
COLLECTION-FREQUENCY-5000	0.69	±	0.04
COLLECTION-FREQUENCY-10000	0.70	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	0.49	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	0.56	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	0.63	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-3000	0.66	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-5000	0.69	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-10000	0.70	±	0.04
DOCUMENT-FREQUENCY-500	0.40	±	0.05
DOCUMENT-FREQUENCY-1000	0.53	±	0.04
DOCUMENT-FREQUENCY-2000	0.63	±	0.05
DOCUMENT-FREQUENCY-3000	0.67	±	0.04
DOCUMENT-FREQUENCY-5000	0.69	±	0.04
DOCUMENT-FREQUENCY-10000	0.70	±	0.04
TERM-FREQUENCY-DOCUMENT-FREQUENCY-500	0.48	±	0.04
TERM-FREQUENCY-DOCUMENT-FREQUENCY-1000	0.56	±	0.04
TERM-FREQUENCY-DOCUMENT-FREQUENCY-2000	0.63	±	0.04
TERM-FREQUENCY-DOCUMENT-FREQUENCY-3000	0.67	±	0.04
TERM-FREQUENCY-DOCUMENT-FREQUENCY-5000	0.69	±	0.04
TERM-FREQUENCY-DOCUMENT-FREQUENCY-10000	0.70	±	0.04
WEIRDNESS-500	0.31	±	0.04
WEIRDNESS-1000	0.33	±	0.04
WEIRDNESS-2000	0.49	±	0.05
WEIRDNESS-3000	0.49	±	0.04
WEIRDNESS-5000	0.54	±	0.07
WEIRDNESS-10000	0.63	±	0.03
WORD-FREQUENCY-500	0.39	±	0.05
WORD-FREQUENCY-1000	0.51	±	0.05
WORD-FREQUENCY-2000	0.62	±	0.05
WORD-FREQUENCY-3000	0.65	±	0.04
WORD-FREQUENCY-5000	0.69	±	0.04
WORD-FREQUENCY-10000	0.70	±	0.04
INFORMATION-GAIN-500	0.53	±	0.04
INFORMATION-GAIN-1000	0.62	±	0.04
INFORMATION-GAIN-2000	0.67	±	0.05
INFORMATION-GAIN-3000	0.70	±	0.04
INFORMATION-GAIN-5000	0.71	±	0.04
INFORMATION-GAIN-10000	0.70	±	0.04
MUTUAL-INFORMATION-500	0.54	±	0.04
MUTUAL-INFORMATION-1000	0.60	±	0.04
MUTUAL-INFORMATION-2000	0.66	±	0.05
MUTUAL-INFORMATION-3000	0.69	±	0.04

Continued on next page

Table B.4 – continued from previous page

Feature selectiom method and feature set size	Macroavg. F_1		
MUTUAL-INFORMATION-5000	0.71	±	0.04
MUTUAL-INFORMATION-10000	0.70	±	0.04
ODDS-RATIO-500	0.53	±	0.04
ODDS-RATIO-1000	0.62	±	0.04
ODDS-RATIO-2000	0.67	±	0.05
ODDS-RATIO-3000	0.70	±	0.04
ODDS-RATIO-5000	0.71	±	0.04
ODDS-RATIO-10000	0.65	±	0.04
CLASS-DISCRIMINATION-MEASURE-500	0.51	±	0.05
CLASS-DISCRIMINATION-MEASURE-1000	0.60	±	0.05
CLASS-DISCRIMINATION-MEASURE-2000	0.60	±	0.05
CLASS-DISCRIMINATION-MEASURE-3000	0.61	±	0.05
CLASS-DISCRIMINATION-MEASURE-5000	0.63	±	0.05
CLASS-DISCRIMINATION-MEASURE-10000	0.68	±	0.05
CHI-SQUARE-500	0.61	±	0.04
CHI-SQUARE-1000	0.64	±	0.04
CHI-SQUARE-2000	0.67	±	0.04
CHI-SQUARE-3000	0.70	±	0.04
CHI-SQUARE-5000	**0.72**	±	0.04
CHI-SQUARE-10000	0.71	±	0.04
NGL-COEFFICIENT-500	0.11	±	0.03
NGL-COEFFICIENT-1000	0.12	±	0.03
NGL-COEFFICIENT-2000	0.13	±	0.03
NGL-COEFFICIENT-3000	0.14	±	0.03
NGL-COEFFICIENT-5000	0.14	±	0.03
NGL-COEFFICIENT-10000	0.15	±	0.03
GSS-COEFFICIENT-500	0.54	±	0.04
GSS-COEFFICIENT-1000	0.61	±	0.04
GSS-COEFFICIENT-2000	0.67	±	0.04
GSS-COEFFICIENT-3000	0.69	±	0.04
GSS-COEFFICIENT-5000	0.71	±	0.04
GSS-COEFFICIENT-10000	0.70	±	0.04
BI-NORMAL-SEPARATION-500	0.60	±	0.04
BI-NORMAL-SEPARATION-1000	0.62	±	0.04
BI-NORMAL-SEPARATION-2000	0.65	±	0.04
BI-NORMAL-SEPARATION-3000	0.65	±	0.04
BI-NORMAL-SEPARATION-5000	0.65	±	0.04
BI-NORMAL-SEPARATION-10000	0.67	±	0.04
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	0.04	±	0.03
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	0.04	±	0.03
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	0.04	±	0.03
CATEGORICAL-PROPORTIONAL-DIFFERENCE-3000	0.05	±	0.04
CATEGORICAL-PROPORTIONAL-DIFFERENCE-5000	0.15	±	0.05
CATEGORICAL-PROPORTIONAL-DIFFERENCE-10000	0.31	±	0.06
DIA-ASSOCIATION-FACTOR-500	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-1000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-2000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-3000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-5000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-10000	0.00	±	0.00

Table B.5: Percentage correctly categorized documents using a Support Vector Machine classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 5 runs.

Feature selectiom method and feature set size	Percent correct		
RANDOM-500	8.58	±	0.50
RANDOM-1000	10.46	±	0.49
RANDOM-2000	18.23	±	0.82
COLLECTION-FREQUENCY-500	71.40	±	0.96
COLLECTION-FREQUENCY-1000	78.43	±	0.92
COLLECTION-FREQUENCY-2000	84.06	±	0.81
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	70.68	±	0.88
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	78.56	±	0.85
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	84.13	±	0.91
DOCUMENT-FREQUENCY-500	66.64	±	1.14
DOCUMENT-FREQUENCY-1000	76.66	±	0.90
DOCUMENT-FREQUENCY-2000	83.16	±	0.76

Continued on next page

Table B.5 – continued from previous page

Feature selectiom method and feature set size	Percent correct		
TFDF-500	70.77	±	0.91
TFDF-1000	79.12	±	0.82
TFDF-2000	84.25	±	0.78
WEIRDNESS-500	38.61	±	1.25
WEIRDNESS-1000	46.30	±	1.29
WEIRDNESS-2000	57.16	±	1.00
WORD-FREQUENCY-500	67.12	±	1.17
WORD-FREQUENCY-1000	76.61	±	0.89
WORD-FREQUENCY-2000	83.22	±	0.76
INFORMATION-GAIN-500	**76.74**	±	0.84
INFORMATION-GAIN-1000	81.39	±	0.76
INFORMATION-GAIN-2000	85.14	±	0.72
MUTUAL-INFORMATION-500	76.68	±	0.70
MUTUAL-INFORMATION-1000	**81.77**	±	0.72
MUTUAL-INFORMATION-2000	85.61	±	0.75
EXTENDED-ODDS-RATIO-500	40.82	±	1.10
EXTENDED-ODDS-RATIO-1000	59.65	±	1.13
EXTENDED-ODDS-RATIO-2000	76.62	±	1.10
CLASS-DISCRIMINATION-MEASURE-500	37.86	±	1.01
CLASS-DISCRIMINATION-MEASURE-1000	46.35	±	0.98
CLASS-DISCRIMINATION-MEASURE-2000	56.53	±	1.15
CHI-SQUARE-500	76.37	±	0.82
CHI-SQUARE-1000	81.11	±	0.66
CHI-SQUARE-2000	84.89	±	0.73
NGL-COEFFICIENT-500	62.68	±	0.74
NGL-COEFFICIENT-1000	67.97	±	0.77
NGL-COEFFICIENT-2000	72.96	±	0.88
GSS-COEFFICIENT-500	76.65	±	0.89
GSS-COEFFICIENT-1000	81.58	±	0.78
GSS-COEFFICIENT-2000	**85.68**	±	0.78
BI-NORMAL-SEPARATION-500	74.35	±	0.81
BI-NORMAL-SEPARATION-1000	79.93	±	0.73
BI-NORMAL-SEPARATION-2000	83.21	±	0.85
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	10.33	±	0.53
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	14.95	±	0.61
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	22.74	±	0.83
DIA-ASSOCIATION-FACTOR-500	7.47	±	0.28
DIA-ASSOCIATION-FACTOR-1000	8.50	±	0.26
DIA-ASSOCIATION-FACTOR-2000	10.11	±	0.21

Table B.6: Macroaverage precision of a Support Vector Machine classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 5 runs.

Feature selectiom method and feature set size	Macroavg. precision		
RANDOM-500	0.19	±	0.29
RANDOM-1000	0.36	±	0.20
RANDOM-2000	0.25	±	0.11
COLLECTION-FREQUENCY-500	0.65	±	0.04
COLLECTION-FREQUENCY-1000	0.70	±	0.04
COLLECTION-FREQUENCY-2000	0.78	±	0.05
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	0.65	±	0.03
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	0.70	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	0.77	±	0.04
DOCUMENT-FREQUENCY-500	0.54	±	0.04
DOCUMENT-FREQUENCY-1000	0.69	±	0.04
DOCUMENT-FREQUENCY-2000	0.78	±	0.04
TFDF-500	0.66	±	0.05
TFDF-1000	0.70	±	0.04
TFDF-2000	0.77	±	0.04
WEIRDNESS-500	0.44	±	0.05
WEIRDNESS-1000	0.49	±	0.06
WEIRDNESS-2000	0.66	±	0.05
WORD-FREQUENCY-500	0.53	±	0.03
WORD-FREQUENCY-1000	0.68	±	0.05
WORD-FREQUENCY-2000	0.77	±	0.04
INFORMATION-GAIN-500	0.67	±	0.04
INFORMATION-GAIN-1000	0.77	±	0.04

Continued on next page

Table B.6 – continued from previous page

Feature selectiom method and feature set size	Macroavg. precision		
INFORMATION-GAIN-2000	0.81	±	0.04
MUTUAL-INFORMATION-500	0.68	±	0.04
MUTUAL-INFORMATION-1000	0.76	±	0.04
MUTUAL-INFORMATION-2000	0.80	±	0.04
EXTENDED-ODDS-RATIO-500	0.38	±	0.04
EXTENDED-ODDS-RATIO-1000	0.52	±	0.05
EXTENDED-ODDS-RATIO-2000	0.69	±	0.04
CLASS-DISCRIMINATION-MEASURE-500	**0.91**	±	0.05
CLASS-DISCRIMINATION-MEASURE-1000	0.89	±	0.04
CLASS-DISCRIMINATION-MEASURE-2000	0.89	±	0.04
CHI-SQUARE-500	0.70	±	0.04
CHI-SQUARE-1000	0.76	±	0.04
CHI-SQUARE-2000	0.80	±	0.04
NGL-COEFFICIENT-500	0.20	±	0.02
NGL-COEFFICIENT-1000	0.24	±	0.03
NGL-COEFFICIENT-2000	0.28	±	0.03
GSS-COEFFICIENT-500	0.68	±	0.04
GSS-COEFFICIENT-1000	0.74	±	0.04
GSS-COEFFICIENT-2000	0.80	±	0.04
BI-NORMAL-SEPARATION-500	0.70	±	0.04
BI-NORMAL-SEPARATION-1000	0.77	±	0.04
BI-NORMAL-SEPARATION-2000	0.80	±	0.04
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	0.68	±	0.47
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	**1.00**	±	0.00
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	**1.00**	±	0.00
DIA-ASSOCIATION-FACTOR-500	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-1000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-2000	0.00	±	0.00

Table B.7: Macroaverage recall of a Support Vector Machine classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 5 runs.

Feature selectiom method and feature set size	Macroavg. recall		
RANDOM-500	0.03	±	0.02
RANDOM-1000	0.04	±	0.02
RANDOM-2000	0.14	±	0.04
COLLECTION-FREQUENCY-500	0.68	±	0.04
COLLECTION-FREQUENCY-1000	0.73	±	0.04
COLLECTION-FREQUENCY-2000	0.80	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	0.67	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	0.73	±	0.04
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	0.80	±	0.04
DOCUMENT-FREQUENCY-500	0.60	±	0.04
DOCUMENT-FREQUENCY-1000	0.73	±	0.04
DOCUMENT-FREQUENCY-2000	0.80	±	0.04
TFDF-500	0.68	±	0.04
TFDF-1000	0.74	±	0.05
TFDF-2000	0.80	±	0.04
WEIRDNESS-500	0.31	±	0.05
WEIRDNESS-1000	0.35	±	0.05
WEIRDNESS-2000	0.60	±	0.06
WORD-FREQUENCY-500	0.60	±	0.04
WORD-FREQUENCY-1000	0.72	±	0.04
WORD-FREQUENCY-2000	0.79	±	0.04
INFORMATION-GAIN-500	0.70	±	0.05
INFORMATION-GAIN-1000	**0.77**	±	0.04
INFORMATION-GAIN-2000	**0.84**	±	0.04
MUTUAL-INFORMATION-500	0.68	±	0.05
MUTUAL-INFORMATION-1000	0.75	±	0.05
MUTUAL-INFORMATION-2000	0.82	±	0.04
EXTENDED-ODDS-RATIO-500	0.40	±	0.06
EXTENDED-ODDS-RATIO-1000	0.59	±	0.05
EXTENDED-ODDS-RATIO-2000	0.74	±	0.05
CLASS-DISCRIMINATION-MEASURE-500	0.36	±	0.05
CLASS-DISCRIMINATION-MEASURE-1000	0.51	±	0.06
CLASS-DISCRIMINATION-MEASURE-2000	0.53	±	0.06
CHI-SQUARE-500	0.67	±	0.05

Continued on next page

Table B.7 – continued from previous page

Feature selectiom method and feature set size	Macroavg. recall		
CHI-SQUARE-1000	0.76	±	0.05
CHI-SQUARE-2000	0.81	±	0.04
NGL-COEFFICIENT-500	0.56	±	0.05
NGL-COEFFICIENT-1000	0.54	±	0.05
NGL-COEFFICIENT-2000	0.54	±	0.06
GSS-COEFFICIENT-500	**0.71**	±	0.04
GSS-COEFFICIENT-1000	0.76	±	0.05
GSS-COEFFICIENT-2000	0.83	±	0.04
BI-NORMAL-SEPARATION-500	0.69	±	0.05
BI-NORMAL-SEPARATION-1000	0.75	±	0.04
BI-NORMAL-SEPARATION-2000	0.79	±	0.04
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	0.01	±	0.01
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	0.06	±	0.03
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	0.11	±	0.03
DIA-ASSOCIATION-FACTOR-500	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-1000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-2000	0.00	±	0.00

Table B.8: Macroaverage F_1-measure of a Naïve Bayes classifier and various feature selection methods with different feature set sizes from the 20-Newsgroups collection, with 10-fold cross-validation averaged over 10 runs.

Feature selectiom method and feature set size	Macroavg. F_1		
RANDOM-500	0.04	±	0.03
RANDOM-1000	0.07	±	0.04
RANDOM-2000	0.17	±	0.04
COLLECTION-FREQUENCY-500	0.66	±	0.03
COLLECTION-FREQUENCY-1000	0.72	±	0.03
COLLECTION-FREQUENCY-2000	0.79	±	0.03
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-500	0.66	±	0.03
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-1000	0.71	±	0.03
COLLECTION-FREQUENCY-INV-DOCUMENT-FREQUENCY-2000	0.79	±	0.03
DOCUMENT-FREQUENCY-500	0.57	±	0.03
DOCUMENT-FREQUENCY-1000	0.71	±	0.03
DOCUMENT-FREQUENCY-2000	0.78	±	0.03
TFDF-500	0.67	±	0.04
TFDF-1000	0.72	±	0.03
TFDF-2000	0.79	±	0.03
WEIRDNESS-500	0.36	±	0.04
WEIRDNESS-1000	0.41	±	0.05
WEIRDNESS-2000	0.62	±	0.04
WORD-FREQUENCY-500	0.56	±	0.03
WORD-FREQUENCY-1000	0.70	±	0.04
WORD-FREQUENCY-2000	0.78	±	0.03
INFORMATION-GAIN-500	0.68	±	0.04
INFORMATION-GAIN-1000	**0.77**	±	0.03
INFORMATION-GAIN-2000	**0.82**	±	0.03
MUTUAL-INFORMATION-500	0.68	±	0.03
MUTUAL-INFORMATION-1000	0.75	±	0.03
MUTUAL-INFORMATION-2000	0.81	±	0.03
EXTENDED-ODDS-RATIO-500	0.39	±	0.04
EXTENDED-ODDS-RATIO-1000	0.55	±	0.04
EXTENDED-ODDS-RATIO-2000	0.71	±	0.03
CLASS-DISCRIMINATION-MEASURE-500	0.52	±	0.06
CLASS-DISCRIMINATION-MEASURE-1000	0.64	±	0.05
CLASS-DISCRIMINATION-MEASURE-2000	0.66	±	0.05
CHI-SQUARE-500	**0.69**	±	0.03
CHI-SQUARE-1000	0.76	±	0.03
CHI-SQUARE-2000	0.80	±	0.03
NGL-COEFFICIENT-500	0.30	±	0.02
NGL-COEFFICIENT-1000	0.34	±	0.03
NGL-COEFFICIENT-2000	0.37	±	0.04
GSS-COEFFICIENT-500	**0.69**	±	0.03
GSS-COEFFICIENT-1000	0.75	±	0.03
GSS-COEFFICIENT-2000	0.81	±	0.03
BI-NORMAL-SEPARATION-500	**0.69**	±	0.04
BI-NORMAL-SEPARATION-1000	0.76	±	0.03
BI-NORMAL-SEPARATION-2000	0.79	±	0.03

Continued on next page

Table B.8 – continued from previous page

Feature selectiom method and feature set size	Macroavg. F_1		
CATEGORICAL-PROPORTIONAL-DIFFERENCE-500	0.02	±	0.02
CATEGORICAL-PROPORTIONAL-DIFFERENCE-1000	0.11	±	0.05
CATEGORICAL-PROPORTIONAL-DIFFERENCE-2000	0.20	±	0.05
DIA-ASSOCIATION-FACTOR-500	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-1000	0.00	±	0.00
DIA-ASSOCIATION-FACTOR-2000	0.00	±	0.00

B.2 The Effect of Globalizing Schemes

Figure B.1 shows the effect of two methods of globalizing the feature values. The Extended Odds Ratio (EOR) is globalized by summarinzing the values from each category, while the Weighted Odds Ratio is globalized by first weighting each feature value by the class size and then summarizing these values. Also shown in the figure is a Weighted Odds Ratio without a logarithm statement, and the Class Discrimination Measure (CDM) which is based on Odds Ratio.

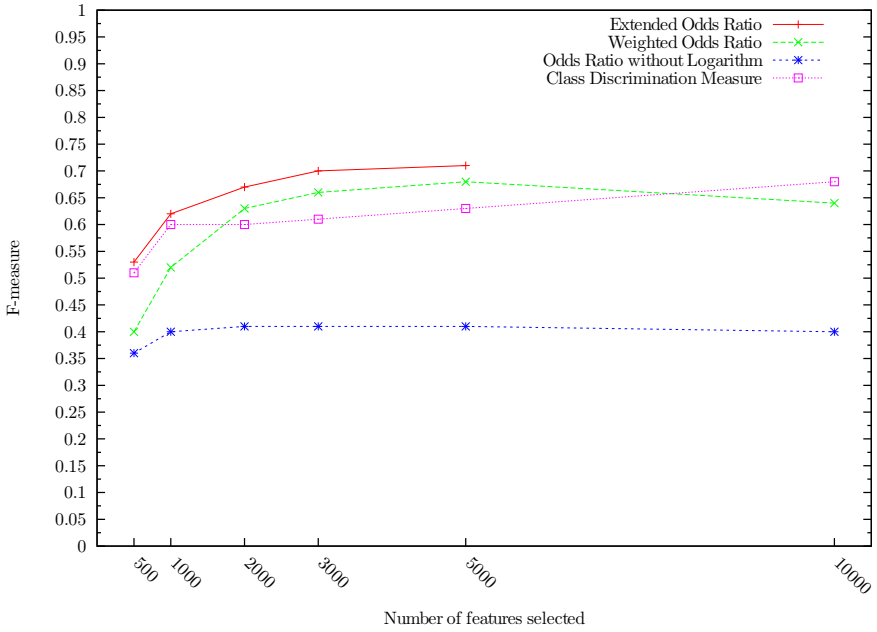

Figure B.1: Naïve Bayes F-measure chart with various feature set sizes from the 20 newsgroups corpus. Various variants of the Odds Ratio metric are compared

B.3 Combination Figures

The following figures shows the results of combination experiments.

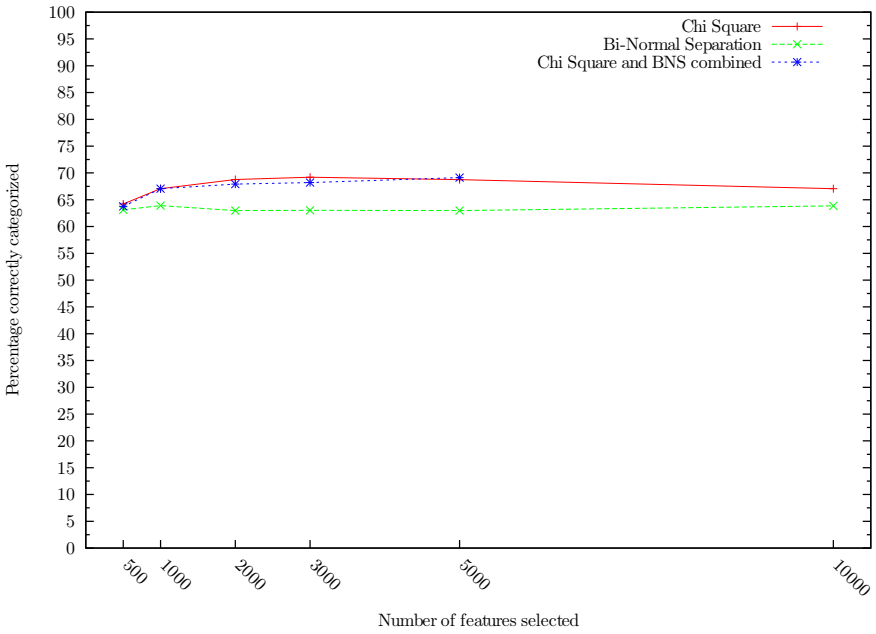

Figure B.2: Percentage of documents correctly classified by a Naïve Bayes classifier using CHI, BNS, and the two combined on the 20 Newsgroups.

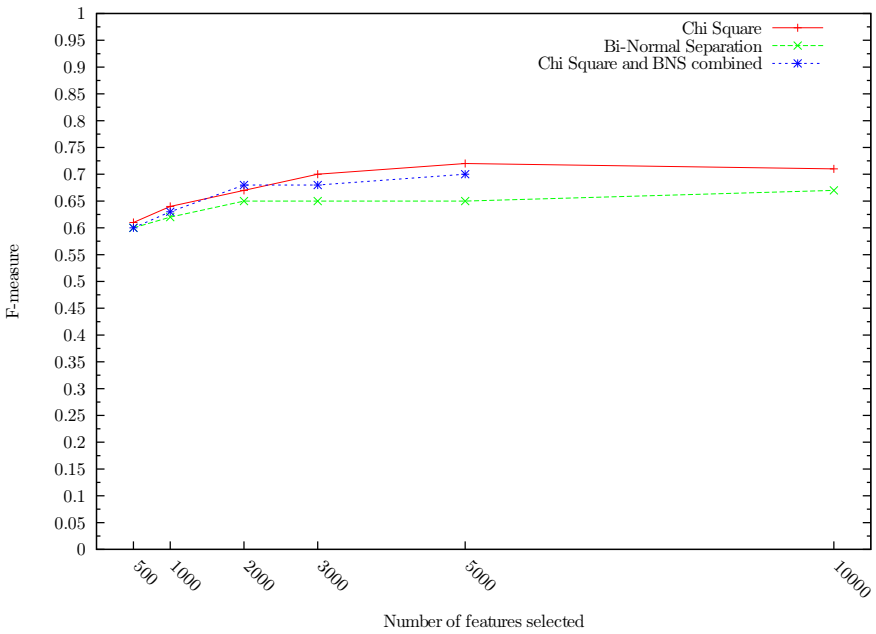

Figure B.3: F-measure for a Naïve Bayes classifier using CHI, BNS, and the two combined at various numbers of features selected from the 20 Newsgroups.

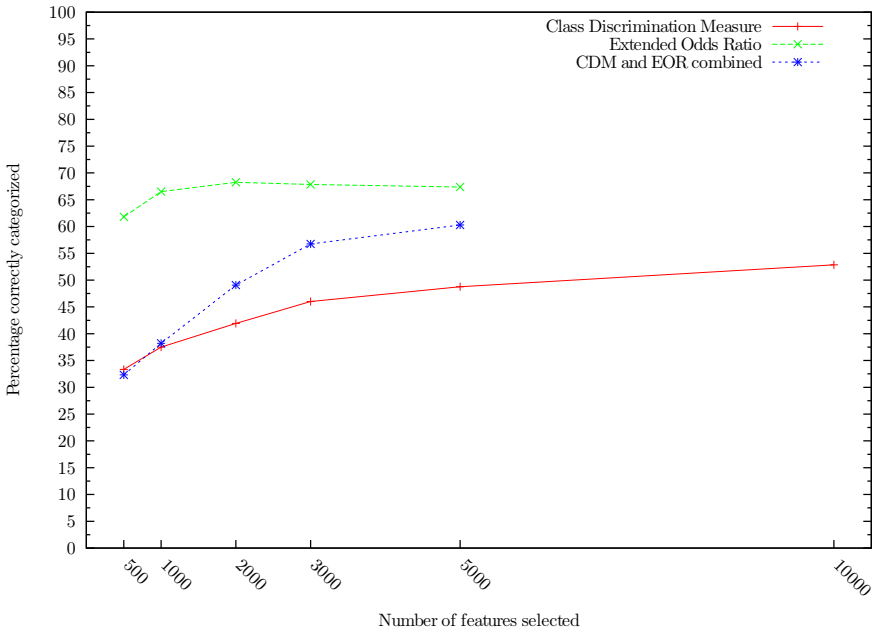

Figure B.4: Percentage of documents correctly classified by a Naïve Bayes classifier using CDM, EOR, and the two combined on the 20 Newsgroups.

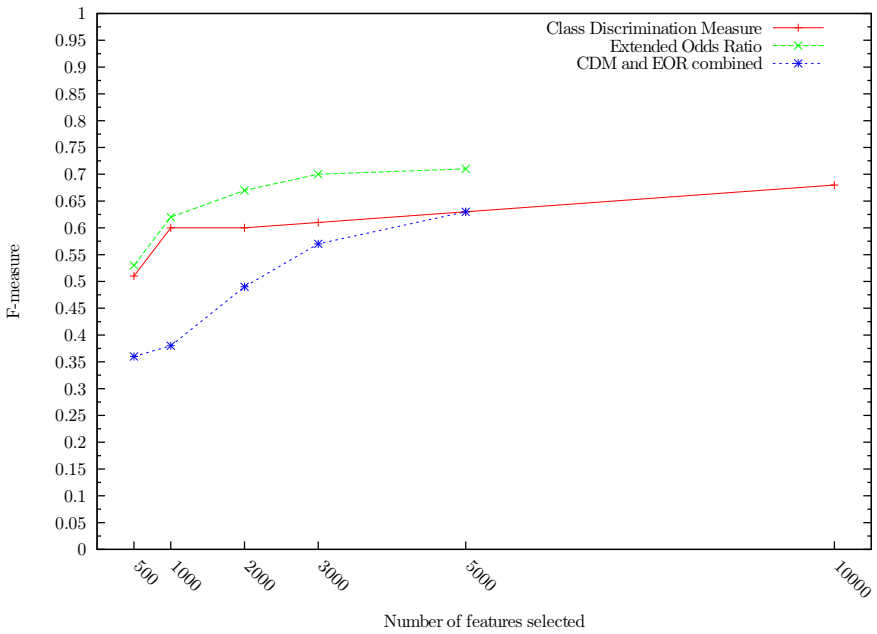

Figure B.5: F-measure for a Naïve Bayes classifier using CDM, EOR, and the two combined at various numbers of features selected from the 20 Newsgroups.

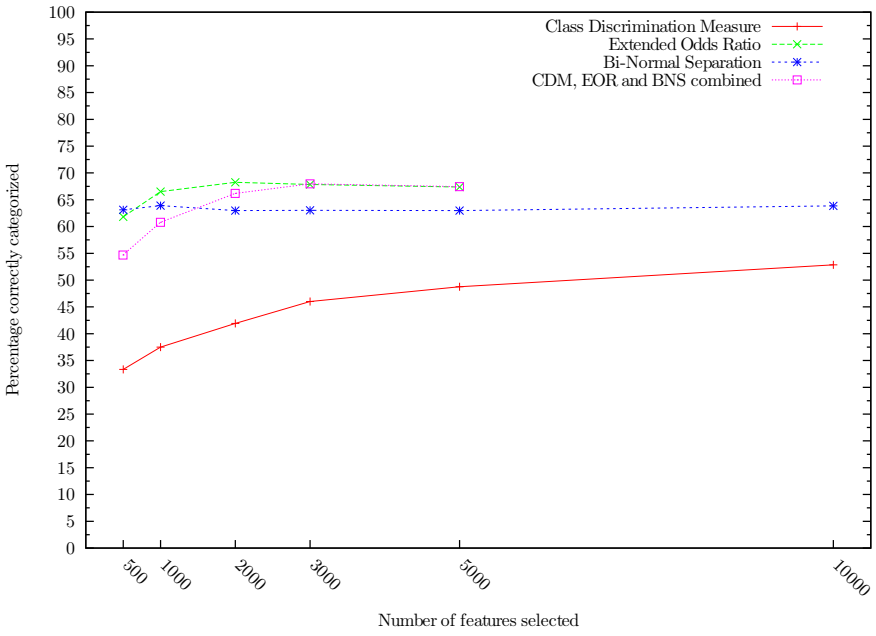

Figure B.6: Percentage of documents correctly classified by a Naïve Bayes classifier using CDM, EOR, BNS, and the three combined on the 20 Newsgroups.

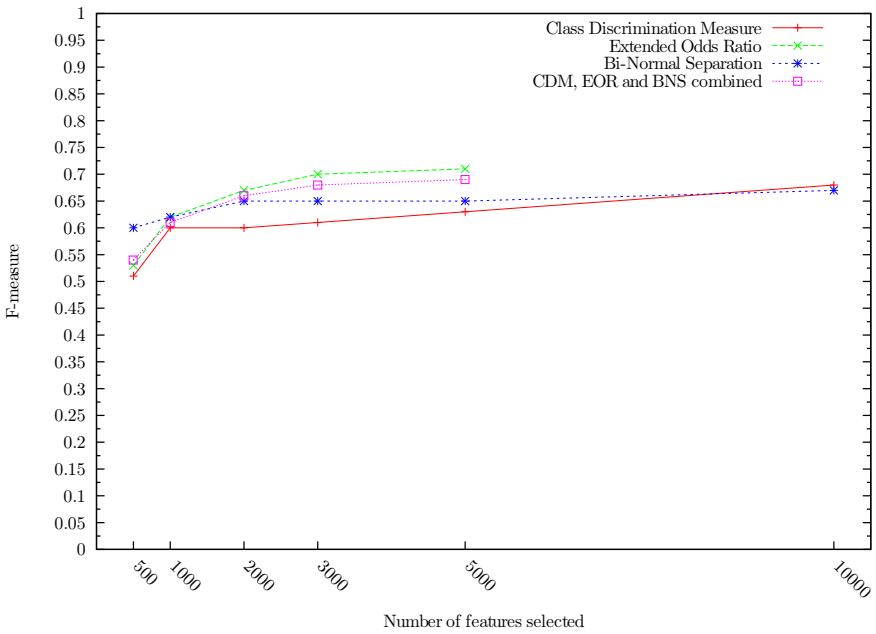

Figure B.7: F-measure for a Naïve Bayes classifier using CDM, EOR, BNS, and the three combined on the 20 Newsgroups.

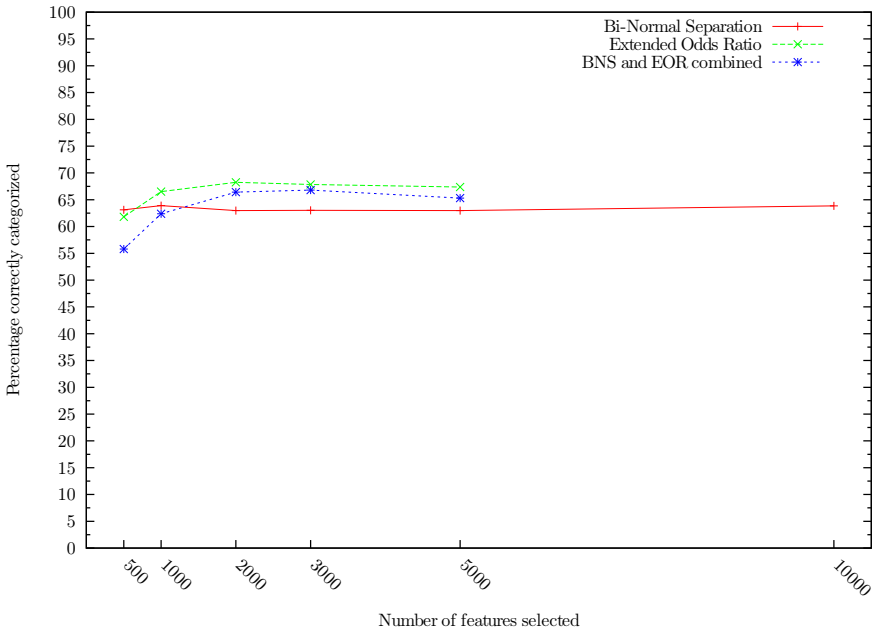

Figure B.8: Percentage of documents correctly classified by a Naïve Bayes classifier using BNS, EOR, and the two combined on the 20 Newsgroups.

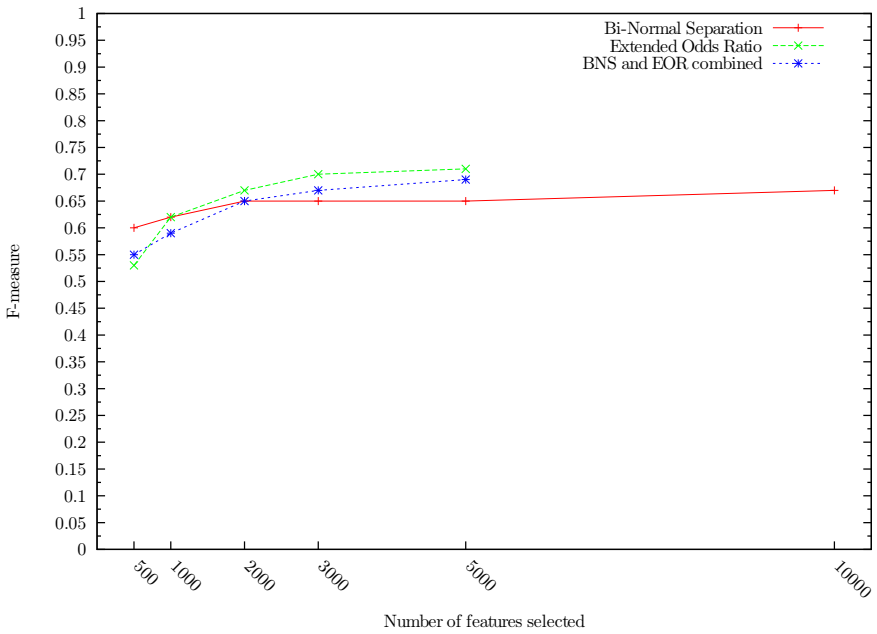

Figure B.9: F-measure for a Naïve Bayes classifier using BNS, EOR, and the two combined at various numbers of features selected from the 20 Newsgroups.

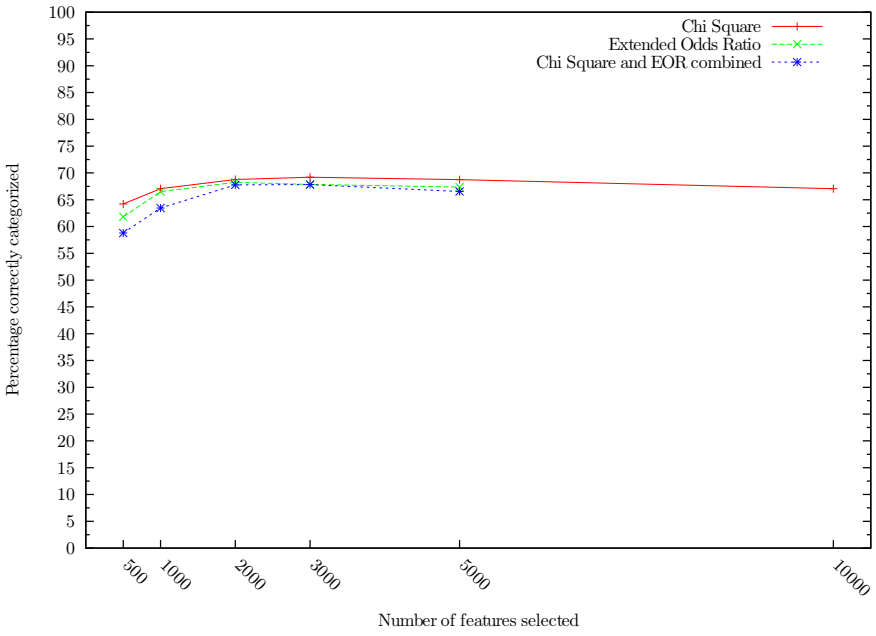

Figure B.10: Percentage of documents correctly classified by a Naïve Bayes classifier using CHI, EOR, and the two combined on the 20 Newsgroups.

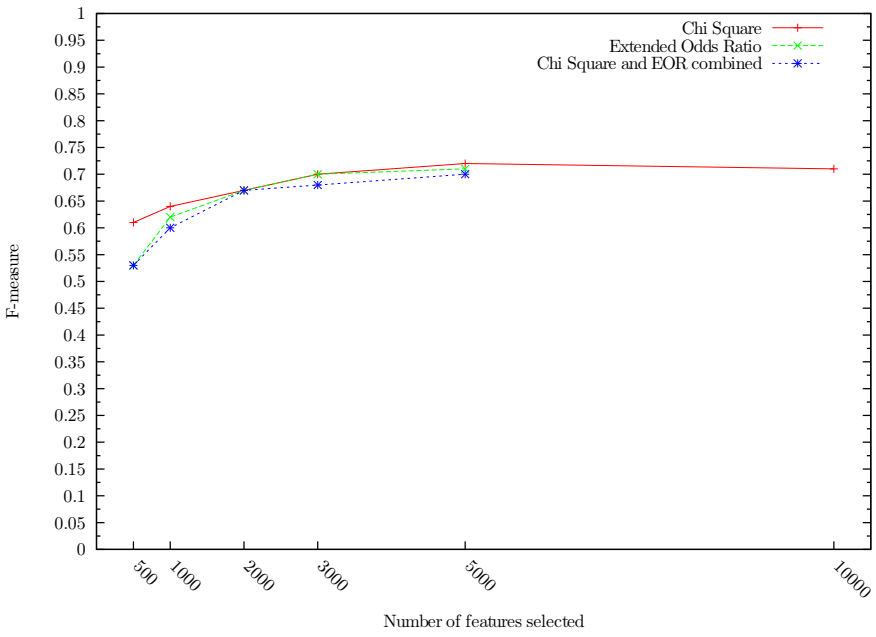

Figure B.11: F-measure for a Naïve Bayes classifier using CHI, EOR, and the two combined at various numbers of features selected from the 20 Newsgroups.

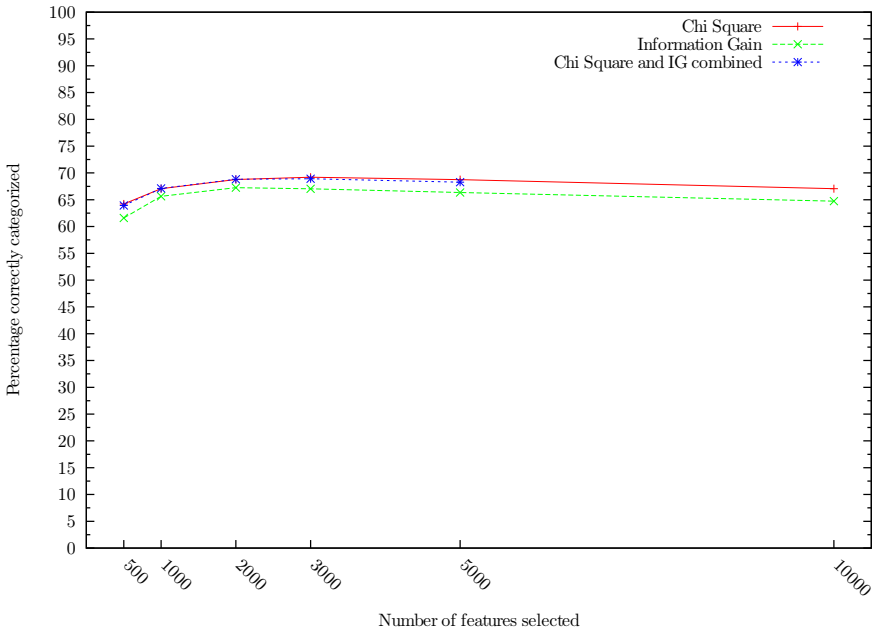

Figure B.12: Percentage of documents correctly classified by a Naïve Bayes classifier using CHI, IG, and the two combined on the 20 Newsgroups.

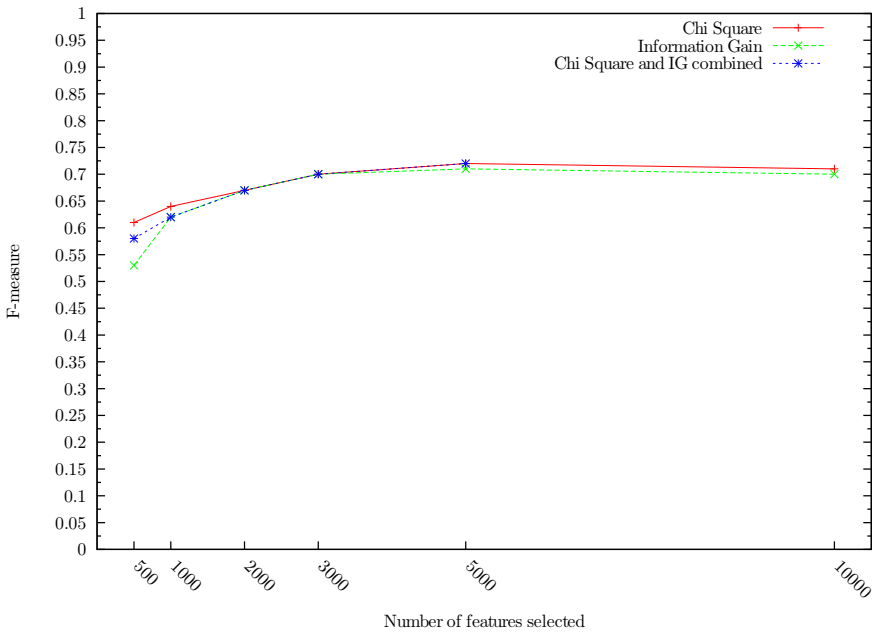

Figure B.13: F-measure for a Naïve Bayes classifier using CHI, IG, and the two combined on the 20 Newsgroups.

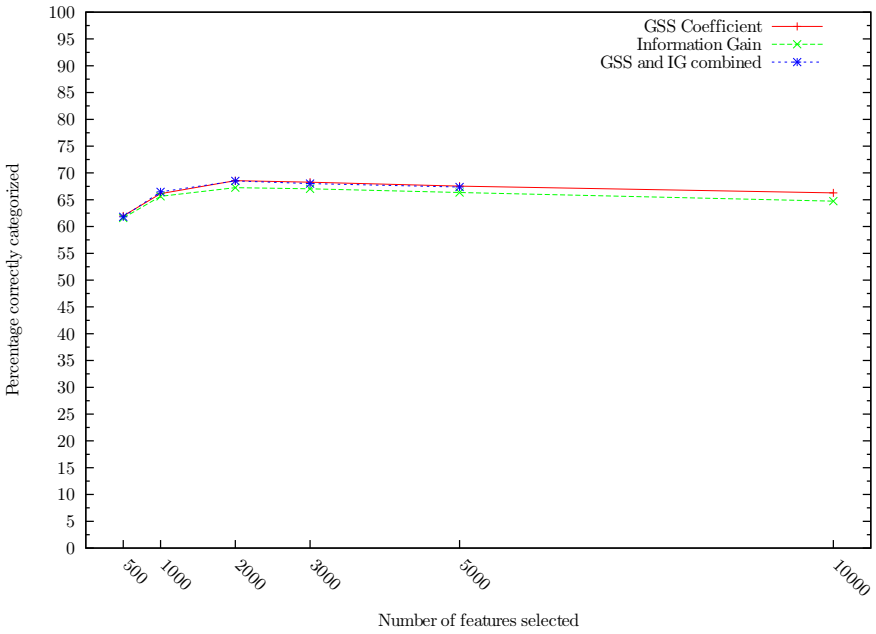

Figure B.14: Percentage of documents correctly classified by a Naïve Bayes classifier using GSS, IG, and the two combined on the 20 Newsgroups.

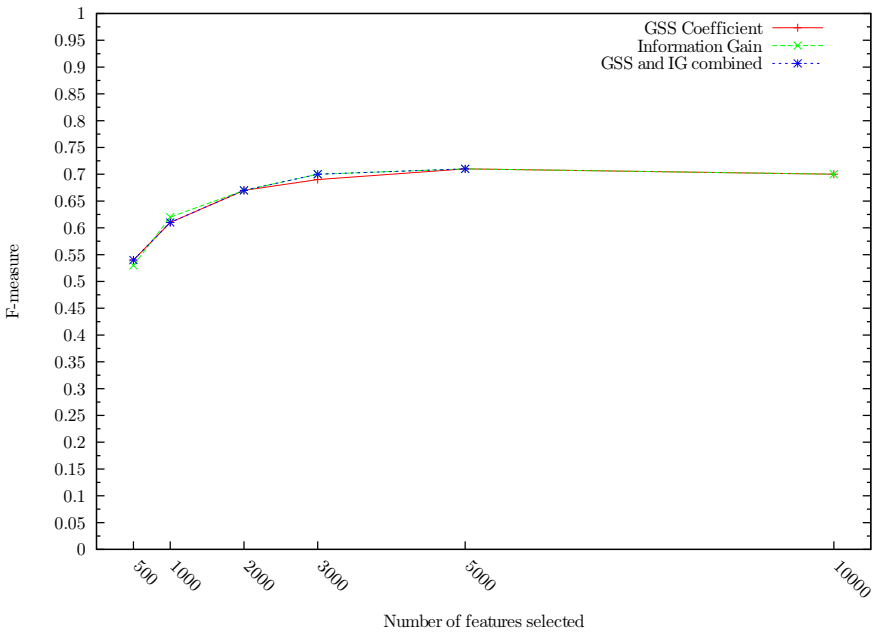

Figure B.15: F-measure for a Naïve Bayes classifier using GSS, IG, and the two combined at various numbers of features selected from the 20 Newsgroups.

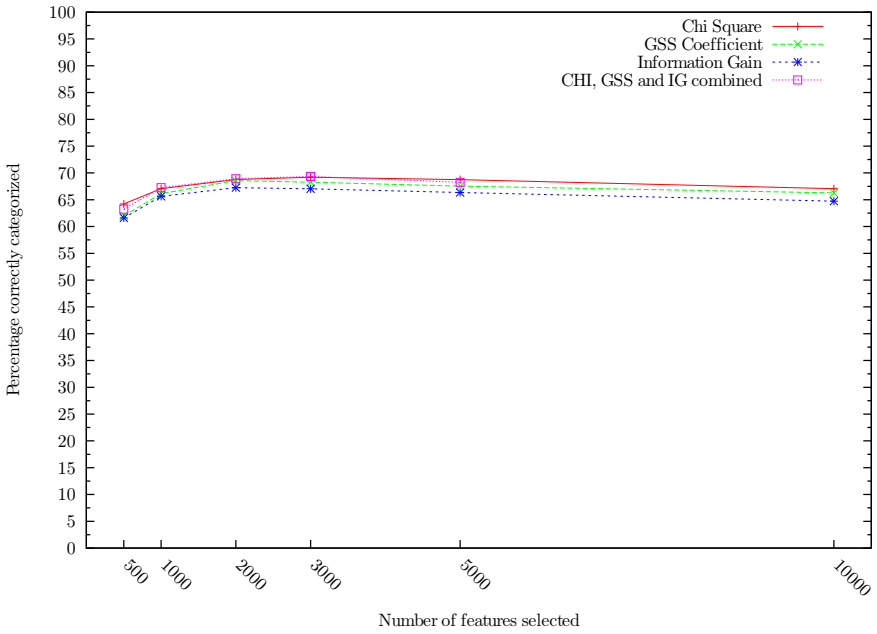

Figure B.16: Percentage of documents correctly classified by a Naïve Bayes classifier using CHI, GSS, IG, and the three combined on the 20 Newsgroups.

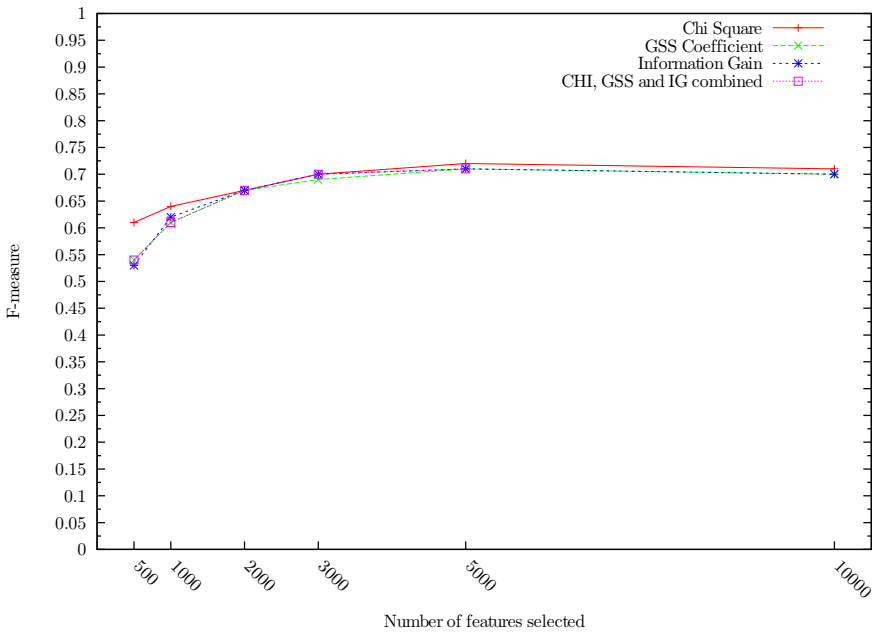

Figure B.17: F-measure for a Naïve Bayes classifier using CHI, GSS, IG, and the two combined at various numbers of features selected from the 20 Newsgroups.

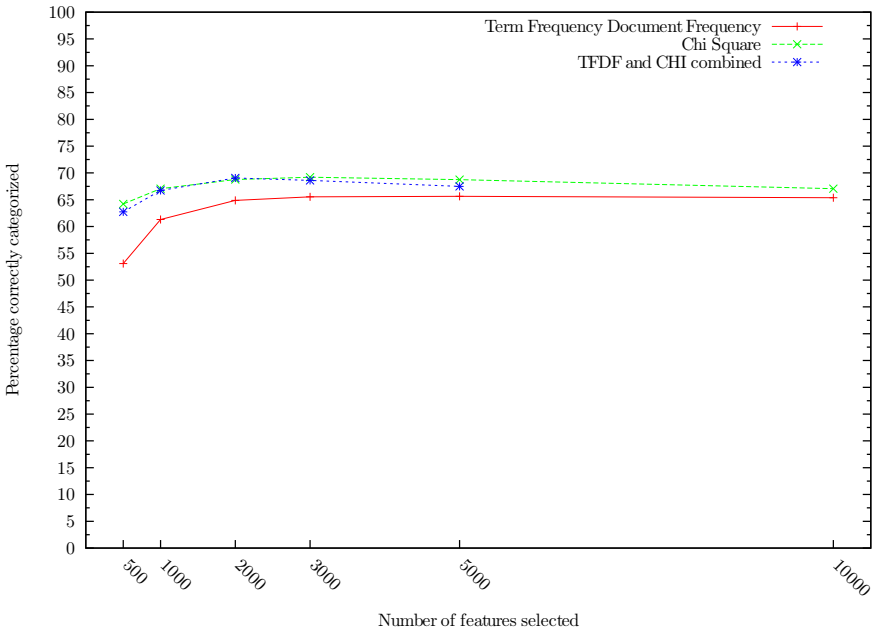

Figure B.18: Percentage of documents correctly classified by a Naïve Bayes classifier using TFDF, CHI, and the two combined on the 20 Newsgroups.

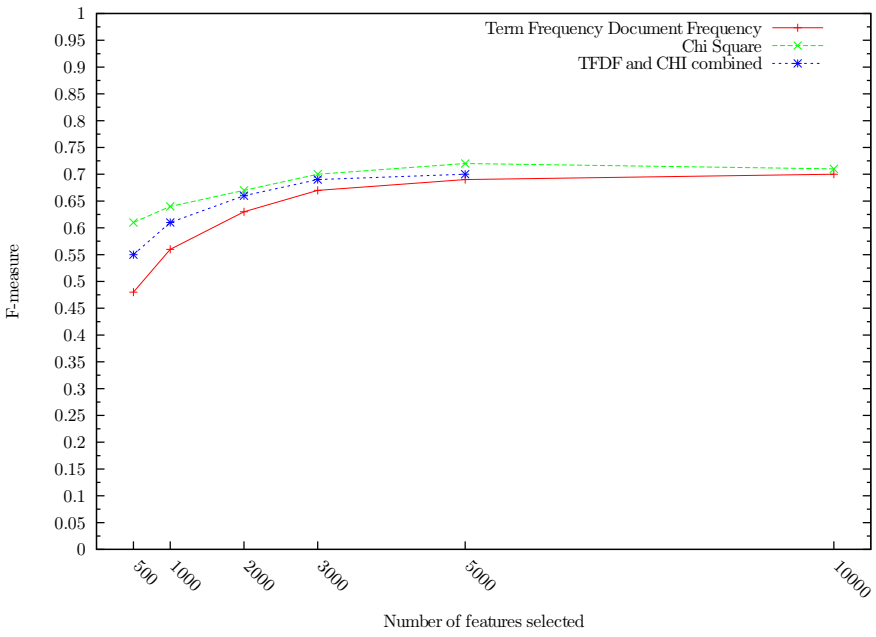

Figure B.19: F-measure for a Naïve Bayes classifier using TFDF, CHI, and the two combined at various numbers of features selected from the 20 Newsgroups.

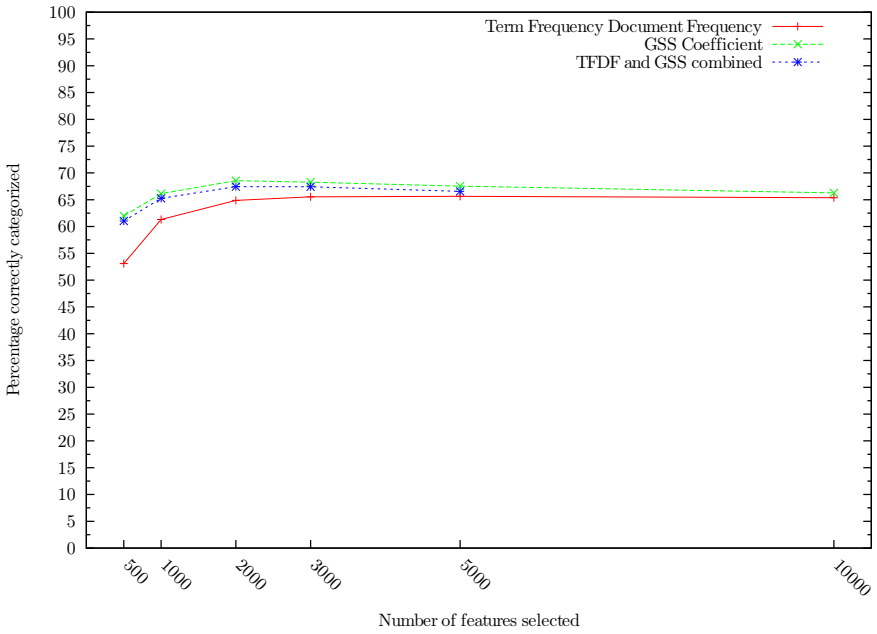

Figure B.20: Percentage of documents correctly classified by a Naïve Bayes classifier using TFDF, GSS, and the two combined on the 20 Newsgroups.

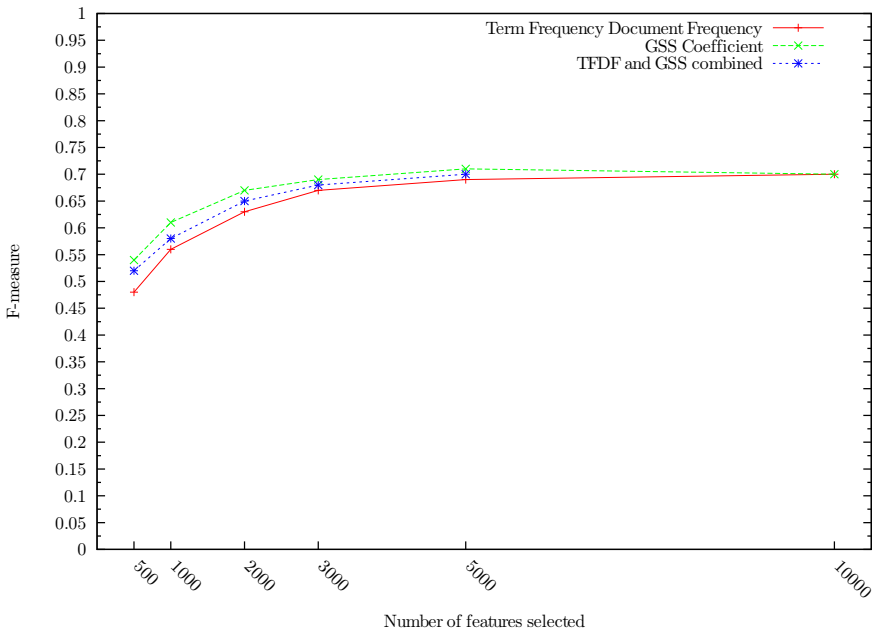

Figure B.21: F-measure for a Naïve Bayes classifier using TFDF, GSS, and the two combined at various numbers of features selected from the 20 Newsgroups.

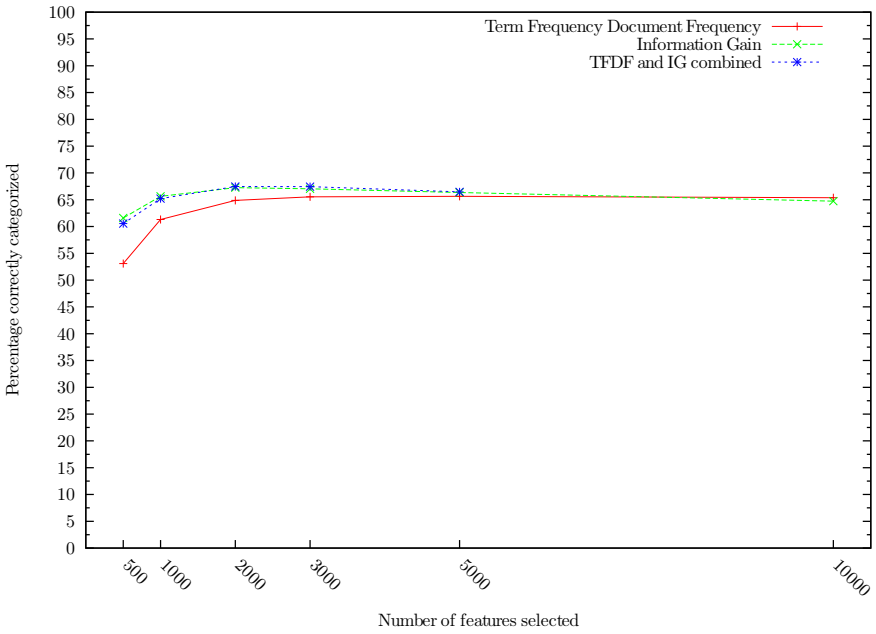

Figure B.22: Percentage of documents correctly classified by a Naïve Bayes classifier using TFDF, IG, and the two combined on the 20 Newsgroups.

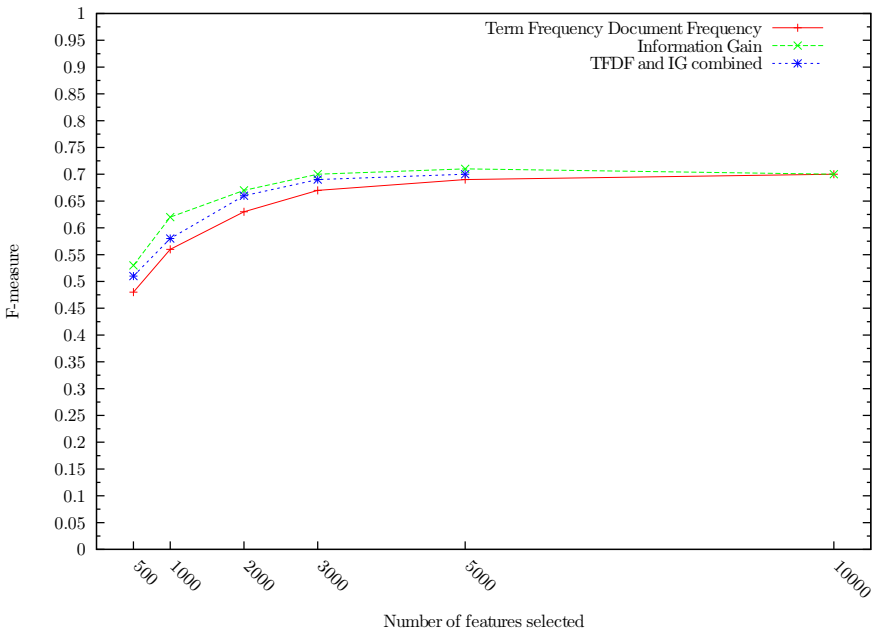

Figure B.23: F-measure for a Naïve Bayes classifier using TFDF, IG, and the two combined at various numbers of features selected from the 20 Newsgroups.

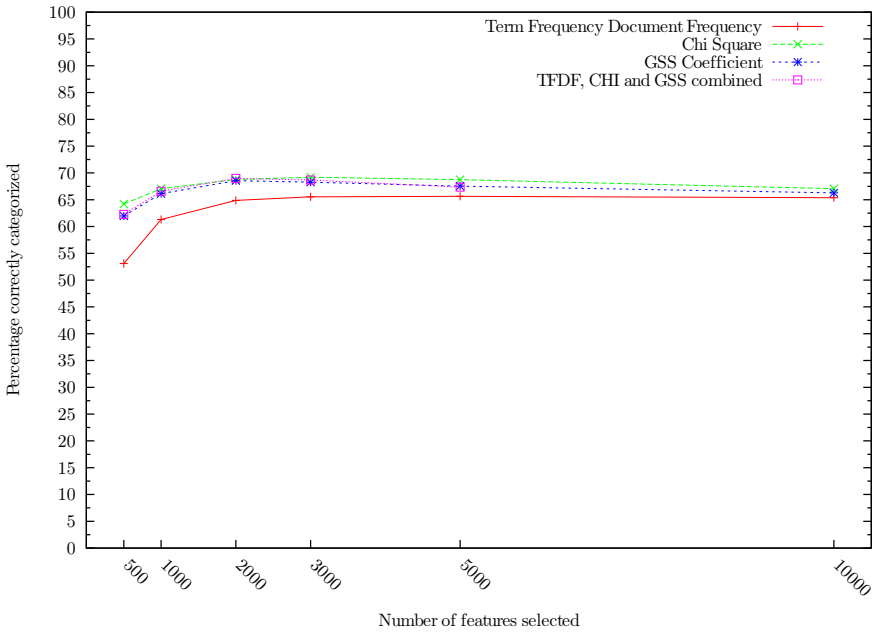

Figure B.24: Percentage of documents correctly classified by a Naïve Bayes classifier using TFDF, CHI, GSS, and the three combined on the 20 Newsgroups.

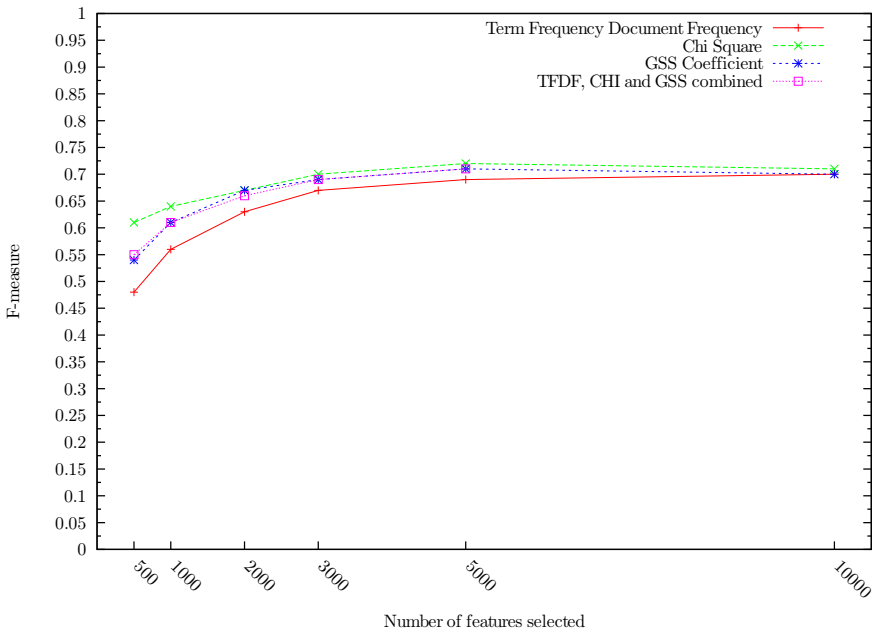

Figure B.25: F-measure for a Naïve Bayes classifier using TFDF, CHI, GSS, and the three combined on the 20 Newsgroups.

Index