SANDIA REPORT
SAND2008-1622
Unlimited Release
March 2008

Yucca Mountain Licensing Support Network Archive Assistant

Justin D. Basilico, Daniel M. Dunlavy, Stephen J. Verzi,
Travis L. Bauer and Wendy Shaneyfelt

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia is a multiprogram laboratory operated by Sandia Corporation,
a Lockheed Martin Company, for the United States Department of Energy's
National Nuclear Security Administration under Contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2008-1622
Unlimited Release
March 2008

Yucca Mountain Licensing Support Network Archive Assistant

Justin D. Basilico
Cognitive and Optical Military Systems
Sandia National Laboratories
P.O. Box 5800 MS1188
Albuquerque, New Mexico 87185-1188

Daniel M. Dunlavy
Computer Science and Informatics
Sandia National Laboratories
P.O. Box 5800 MS1318
Albuquerque, New Mexico 87185-1318

Stephen J. Verzi , Travis L. Bauer, and Wendy Shaneyfelt
Cognitive Systems Research and Applications
Sandia National Laboratories
P.O. Box 5800 MS1011
Albuquerque, New Mexico 87185-1011

Abstract

This report describes the Licensing Support Network (LSN) Assistant—a set of tools for categorizing e-mail messages and documents, and investigating and correcting existing archives of categorized e-mail messages and documents. The two main tools in the LSN Assistant are the LSN Archive Assistant (LSNAA) tool for re-categorizing manually labeled e-mail messages and documents and the LSN Real-time Assistant (LSNRA) tool for categorizing new e-mail messages and documents. This report focuses on the LSNAA tool.

There are two main components of the LSNAA tool. The first is the Sandia Categorization Framework, which is responsible for providing categorizations for documents in an archive and storing them in an appropriate Categorization Database. The second is the actual user interface, which primarily interacts with the

Categorization Database, providing a way for finding and correcting categorizations errors in the database.

A procedure for applying the LSNAA tool and an example use case of the LSNAA tool applied to a set of e-mail messages are provided. Performance results of the categorization model designed for this example use case are presented.

ACKNOWLEDGMENTS

We thank Gordon Appel for providing guidance and feedback on this project from beginning to end. We thank Gordon Appel and Jerry Weiser for providing data used in developing the LSNAA tools.

This page intentionally left blank.

CONTENTS

FIGURES

TABLES

NOMENCLATURE

ART	Adaptive Resonance Theory
DOE	Department of Energy
LL	Lead Laboratory
LSN	Licensing Support Network
LSNAA	Licensing Support Network Archive Assistant
LSNRA	Licensing Support Network Real-time Assistant
ML	Machine Learning
SME	Subject Matter Expert
SNL	Sandia National Laboratories

1. INTRODUCTION

The Licensing Support Network (LSN) Assistant application is a set of tools for categorizing new e-mail and documents, and investigating and correcting existing archives of categorized e-mail and documents. The two main tools in the LSN Assistant are the LSN Archive Assistant (LSNAA) tool for categorizing existing e-mail and documents and the LSN Real-time Assistant (LSNRA) tool for categorizing new e-mail and documents. This document describes the LSNAA tool; a subsequent report will document the LSNRA tool.

There are two main components of the LSNAA tool. The first is the Sandia Categorization Framework, which is responsible for providing categorizations for all the documents in an archive and storing them in an appropriate Categorization Database. The second is the actual user interface, which primarily interacts with the Categorization Database, providing a way for finding and correcting categorizations errors in the database.

1.1. Problem Description

Sandia's Licensing Assessment and Technical Analysis organization (06853) is responsible for developing the LSN for Yucca Mountain as required by the Nuclear Regulatory Commission (NRC). The LSN is a web application designed to capture all licensing-relevant documents, including emails and any attachments thereto.

To achieve the LSN objective of capturing all licensing-relevant documents, the Lead Laboratory (LL) relies on subject matter experts (SMEs) to first determine whether a document is "relevant" or "not-relevant", then to categorize the relevant ones as either "privileged" or "not privileged". The Department of Energy (DOE) has provided detailed guidance on how to make these categorizations (McRae, 2005; Otis, 2003).

The document data set is substantial in size as it includes all incoming and outgoing e-mail messages and their attachments by all individuals working in LL activities. Individuals working on LL activities are responsible for determining the relevance and categorization of their own incoming and outgoing email messages and attachments following the guidelines set by the DOE. However, the individual biases of these email reviews lead to widely inconsistent categorization decisions. Therefore, to ensure that all emails and attachments are properly and consistently categorized, the LL SMEs perform a final review of each email and attachment.

The LSNAA tool is intended to significantly increase the efficiency of the review and categorization process in terms of schedule and cost. The LSNAA tool scans document sets taken from archives and generates assessment reports for the SMEs to evaluate. Cognitive models, referred to throughout this document as *categorizers*, representing the categorizations made by real SMEs are used to prioritize a review of the document sets. Assessment reports generated by the LSNAA tool provide real SMEs with enough "at-a-glance" data to decide to examine a selective document set further or forego an in-depth study. The reports include the categorizations of documents in reviewed sets according to their relevance, excluded classes, and privilege status. The consistency and quality of the evaluation is expected to increase with the

use of the LSNAA tool, as the categorizers are tuned to the categorizations provided by the SMEs.

1.2 Data Requirements

In order to build an accurate document categorization system for the LSNAA tool, a representative dataset of example document categorization must be provided. Listed below are the data (document) requirements we used in building categorizers for use in the LSNAA tool.

Coverage – The document set as a whole must be representative of the actual documents that are expected to be categorized by the system. In order to be accurate, the system must be trained on documents that are similar to those it is expected to categorize.

Accuracy – Each document must have an accurate categorization assigned to it. The categorizer can only be as accurate as the example classifications that it is provided.

Quantity – The risk of building an accurate classifier is lowered to an acceptable level if there are at least 20,000 documents in each document class. Half of these documents will be used for training and half will be withheld for validation. Since the classifier will be as good as the data it is trained on, a classifier can still be built from less data, but the more example documents that are given, the better the classifier will perform.

Distribution – The relative number of documents in each class must match the relative distribution of documents among the class. For example if there are two classes, "relevant" and "non-relevant", and in general 40% of the documents are judged relevant and 60% are non-relevant then we would want at least 20,000 "relevant" documents and 30,000 "non-relevant" documents.

Supplemental Data – It is not required, but some approaches to classification can make use of supplemental document sets in order to improve performance. It would probably be the most helpful in the case that the requirements for the number of accurate classifications provided cannot be met. There are several options for what could be contained in the supplemental data:

- These documents could have classifications that either have not been verified as accurate or are known as not being completely accurate.
- These documents could be ones with classifications that are not as representative of the whole of the documents that are expected to be classified by the system.
- These documents could be ones that are representative of the documents that are expected to be classified by the system but do not have classifications assigned to them.
- If provided, it would be important to distinguish this data from the other training data.

Details – The following are the requirements for the attributes of the data provided:

- A file explaining of the origin and format of the document set along with an explanation of each of the classes that can be assigned to a document.
- For each email-type document, the following information must be provided:

- Subject – The subject field of the email.
- Date – The date and time that the email was sent.
- From – The email address of the sender of the email.
- To – The email addresses of the recipients of the email. (The additional Carbon-copy (CC) and blind-carbon-copy (BCC) recipients may also be included.)
- Text – The actual text of the document
- Classification(s) – The classification(s) assigned to the document.

- For each classification for the document the following information must be provided:
  - Classifier – An identifier for the classifier, such as an email address.
  - Class(es) – The class(es) assigned.
  - Date – The date of the classification.

1.3 Software Requirements

The goal of the LSNAA is to significantly ease and accelerate the review of documents for licensing relevance and categorize relevant ones according to their privilege status based on example data provided by the Lead Laboratory. A set of software requirements has been created to address the needs of SMEs and the Lead Laboratory. Appendix A presents the complete list of software requirements. Note that the list of requirements is for the both the LSN Archive Assistant and Real-time Assistant applications, since they are so closely related in scope and function.

1.4 Solution Process

For a given set of documents to be reviewed and possibly re-categorized, we have developed the following solution process.

1. Assemble a small validated, labeled set of documents from the entire set.
2. Build a set of potential categorizers using several algorithms and the validated set.
3. Identifying the highest performing algorithms applied to the validated set.
4. Build a categorizer from the best performing algorithms.
5. Tune the categorizer to obtain optimum performance on the validated set.
6. Categorize all documents in the entire set.
7. Have SMEs manually review and re-categorize the documents for which the system categorizations are ambiguous (i.e., those documents for which the system was unsure of the categorizations).
8. Build new categorizers (as in steps 2-5) using the set of documents validated by the SMEs.
9. Categorize all documents in the entire set.
10. Determine if more work is needed to certify the current categorizations as final by assessing any discrepancies between the current and previous system categorizations.
11. Repeat steps 7-10 until satisfied with the results of the system categorizations.

We present in this report an example of this process for the set of e-mail messages from the TSPA group at Yucca Mountain associated with Condition Report (CR) 9601. This set of e-mail

was identified as potentially being incorrectly categorized by some of the creators and recipients. This was the motivation behind the development of the LSNAA tool.

2. THE LSNAA GRAPHICAL USER INTERFACE

The LSNAA application is a graphical user interface for examining an email archive to find and fix incorrect email categorizations. The user interface is designed to make use of automated categorizations created using the Sandia Categorization Framework to assist the user in finding emails that are likely to be incorrectly categorized.

2.1 Software Requirements

The LSN Assistant system, with its main user interface through the LSNAA, was created to conform to a set of requirements that were gathered from the customer and potential users. The requirements are listed in Appendix A.

2.2 Interface Design

From a usability perspective, the LSNAA user interface (as seen in Figure 1) is designed to be similar to a typical email program such as Microsoft Outlook. The reason for designing the layout of the interface in this manner is to make it consistent with the user's normal work flow so that it can be learned quickly. The visual layout of the application follows a three-panel approach. It contains a panel for specifying search criteria for email, a panel that displays a list of email, and a panel for displaying the content of a selected email.

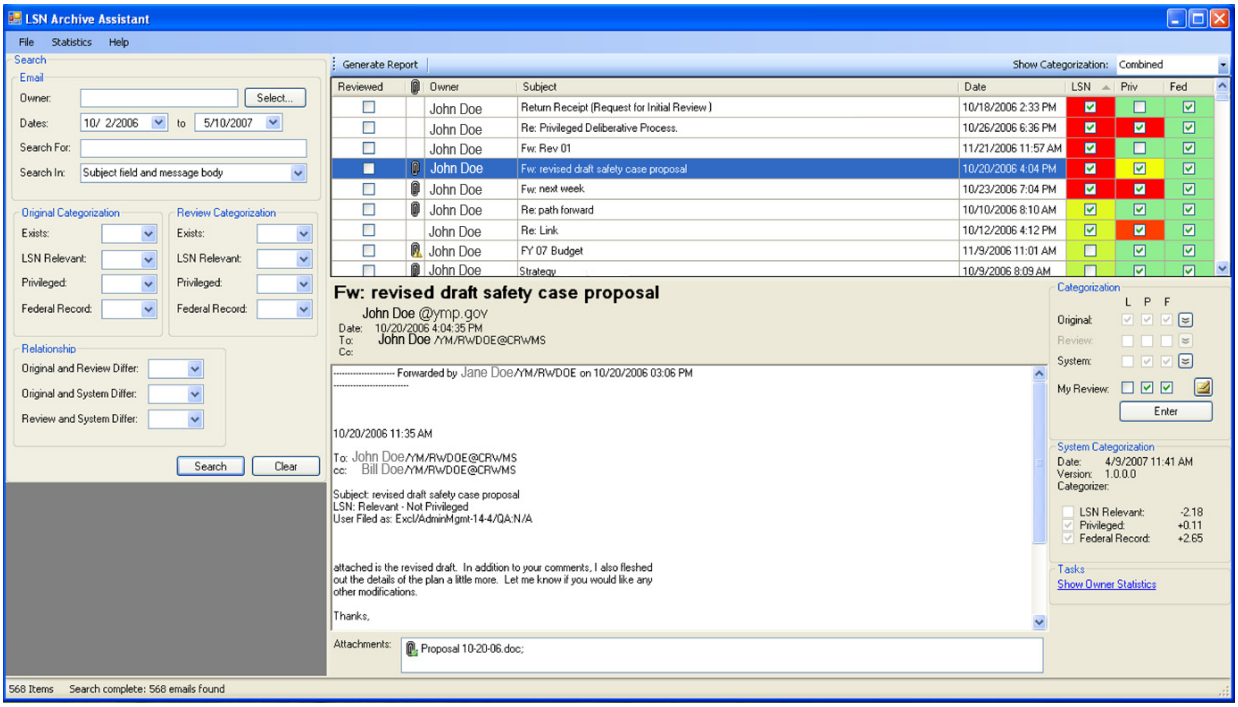

Figure 1. The LSNAA Graphical User Interface.

Since the tool is focused on categorization, the program presents a view of the data regarding three possible categorizations for each email:

- *Original* – The Original categorization is the categorization provided by the owner of the email (the person whose inbox the email belongs to).
- *Review* – The Review categorization is the most recent categorization provided by a certified LSN email reviewer.
- *System* – The System categorization is the most recent categorization provided by the LSN Assistant system's automated categorizer.

It also maintains the Current categorization, which is the Review categorization if it exists, and if not it is the Original categorization. Typically, an email will have an Original and System categorization, only emails that have been reviewed using the tool have a Review categorization. The program is also implemented to handle missing Original and System categorizations.

2.2.1 Search for Information

Because the target usage of the LSNAA is in large email archives, the user interface includes a large search panel. The search panel contains many different fields that allow a user to construct powerful search criteria in order to find the appropriate emails. In addition to searching standard meta-data such as the email owner and the received date, it allows for text searches of the textual subject, body, and attachment fields of an email. The search also allows the user to select specific categorizations from the Original, System, and Review regarding the status of whether or not the categorization exists and if it is marked as LSN Relevant, Privileged, or Federal Record. There is also the option to search based on differences between the Original, System, and Review categorizations to make it easy to find divergent categorizations. All of the search criteria are combined together in the form of a large logical "and" statement; specifying additional criteria constrains the search. Providing such a capability allows the user to find and review specific emails of interest from a large dataset.

2.2.2 Presentation of Information

After a search is performed, one of the main benefits of the LSNAA is in the email list panel. The panel displays some of the meta-data of all email that satisfy the search criteria in a tabular format. The email list can be sorted by any of the columns in the tabular format. The table includes the typical information to identify an email through the owner name, subject text, received date, and an icon indicating the existence of attachments. The real utility of the email list comes in the columns displaying the categorization status of each email. Any of the Original, Review, and System categorizations can be displayed for an email. To display a categorization, three columns are used, one for each category: LSN Relevant, Privileged, and Federal Record. A check box in each of these columns indicates whether the categorization has assigned the email to each of the categories. When a categorization does not exist, no check box is displayed in any of the three columns.

The most useful display for the categorization status is the Combined display that digests the three different categorization types into a single display. It uses the check boxes to display the

Current categorization (Review if it exists; otherwise Original), similar to the other categorizations. However, it also displays a different background color for each cell to draw the user's eye to potentially incorrect categorizations. When the System categorization disagrees with the Original categorization for a particular category, a shade of red is displayed in the background of the cell to indicate the absolute value of the difference: a darker shade of red indicates a larger certainty in the System's categorization. In a similar fashion, green is used to indicate that the System's categorization agrees with the Original categorization. When the System is unsure of the categorization, the background color is set to yellow. In addition to the background colorings, each of the columns for the three categories can be sorted according to the value of the difference between the Original and System categorizations. This allows a user to sort the results of a search to prioritize the review of categorizations with a higher likelihood of being incorrect. If a Review exists for an email, then the Review categorization is displayed in the checkboxes with a blue background to distinguish that it has already been reviewed.

The email panel displays the content of a single selected email and contains the controls for providing a new categorization for an email. The display of the email includes standard fields, including the subject, from, date, to, and cc fields. It also includes a list of files that are attached to the email, which can be opened using the application. The control for entering a categorization allows the user to select check boxes to indicate if an email belongs to each of the LSN Relevant, Privileged, and Federal Record categories, plus an optional comment field. There are also short-cut buttons for filling the review fields from the Original, Review, and System categorizations, if they exist.

One of the requirements of the LSNAA is the ability to read and categorize attachments. As part of the email processing and automated categorization, this means that the system attempts to extract text from attached files in known file types. The status of extraction is displayed in the user interface both in the email list and in the email display window. The program displays a special icon in the email list and on the email itself in the case when an email has an attachment that text could not be extracted from in order to indicate to the user that the system could not use that attachment in its automated categorization. This is done to inform users that the system may not be using all available information and that a human inspection of these attachments may be needed.

In addition to the main search and categorize functionality, the user interface also offers the ability to look at statistics regarding the email of a user or the archive as a whole. This includes the number of email in each category plus the number (and percentage) of email that have an Original, System, Review, or Current categorization. This can be used to monitor the overall quality of an email archive with respect to the number of discrepancies between the current categorizations and the system categorizations plus the amount of the archive that has been reviewed.

2.3 Process Pipeline

The LSNAA represents a front-end for the LSN Assistant system by interacting with an email categorization database. This database contains all of the relevant information about an email including its content, meta-data, attachment text, and categorizations. Because the front-end user

interface just handles the display of information in the database and the entry of new human categorizations for existing email, a set of back-end tools is used to populate the rest of the information.

This set of back-end tools provides a processing pipeline for getting an existing email archive into the database and providing system categorizations. Figure 2 gives a conceptual presentation of the different parts of the pipeline and the data that flows between each component.

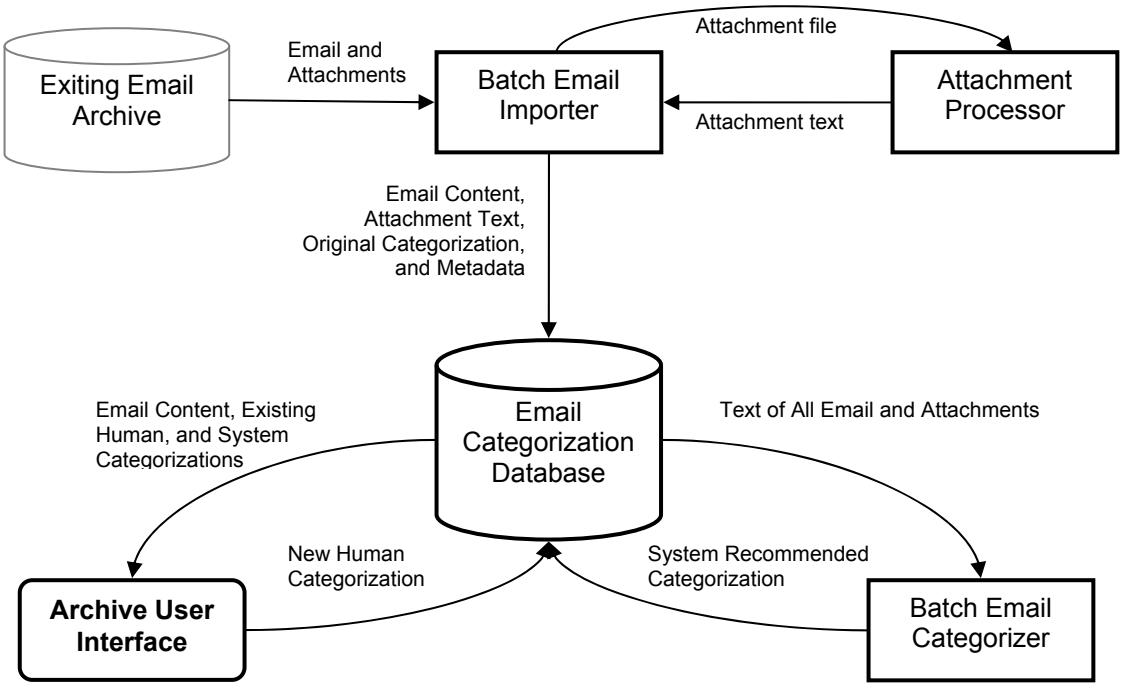

Figure 2. The LSNAA System.

To start the processing pipeline, an existing archive of email must be selected to be imported into the database. In the case of the TSPA email review, the existing email data was contained in a Lotus Notes database. To begin processing the information, the emails (and attachments) were extracted from the Lotus Notes database. Each email was converted to an XML representation containing the metadata plus the body of the email. Each attachment was extracted and saved in its native format. Once the emails have been converted to this XML format, the Batch Email Importer is able to read in the XML files and add them to the new email categorization database. This includes processing the email metadata, body, and determining the Original categorization of the email (if any). It also attempts to extract text from attachments that have a known set of file types (Microsoft Office, PDF, and plain text) and inserts the extracted text into the database. After the batch import is complete, all of the necessary information for the review and automated categorization of the documents is contained in the database.

To provide the System's recommended categorizations for the email, the Batch Email Categorizer is run against the database. This program retrieves the text of each email and its attachments and uses its automated categorizer to create a new System categorization for each email. These new System recommended categorizations are then recorded in the database along with a record of the date of categorization, the version of the software used, and the version of the categorizer used. When a new system categorizer is to be created, the Categorizer Builder retrieves the emails that have been reviewed in the database along with their text, attachment text, and Review categorization and passes them to the machine learning algorithms to build the categorizer.

2.4 Database Interactions

From a technical perspective, the LSNAA user interface provides a front-end to a database of email and categorizations. It allows a user to search the database for emails of interest, view the email with their categorizations, and provide new categorizations when necessary. The application supports two ways of connecting to an email archive database: through direct connection to an email database or through a connection to an email archive web service, that acts as a proxy for the database.

2.4.1 Categorization Used to Prioritize Data

A key advantage of the LSNAA application is the ability to use the categorizations generated automatically by the LSN Assistant system to prioritize which emails to review. These System categorizations are used in two key ways: in the search for email and in the display of search results.

For search, the most recent System categorization for each email can be used as a search criterion. When used in conjunction with the other search criteria, this allows a user to find emails where the System categorization differs from the Original or Review categorizations either in selected categories or in any category. This allows a user to limit their searches to look at email where the System believes there is the possibility of an incorrect categorization.

Beyond the search criteria, the System categorization for each email is used in the display of the results of an email search. The display includes both the ability to show the System's recommended categorization and the ability to show the discrepancy between the System's categorization and the Original categorization.

In addition, because the email search results allow email to be sorted by any of the columns displayed, the emails can be sorted according to their discrepancy. This allows a user to start at the top of the list with the email where the System categorization is the most certain in its difference from the Original categorization. When there are limited resources available for reviewing email, this feature allows a reviewer to focus on areas of the data where the system identifies the greatest differences in categorizations.

2.4.2 User Decisions and Database Updates

All user actions regarding the categorization of emails are stored in the database. This is done to provide a history of all actions regarding an email. The display of Original, System, and Review categorizations are a view of the data that is updated when new categorizations are entered into the database. The Review categorization is updated when a user enters a new review for an email. The System categorization is updated when a new version of the automated categorizer runs against the database. This provides a full trace of the entire categorization history for an email, including which System recommendations are associated with the different Review categorizations.

In addition to providing a review categorization for an email, a user can also mark an email as "Evaluated". This "Evaluated" flag indicates that someone has looked at some aspect of the email (usually the subject line), but has not conducted a full review, and has indicated that it does not need to be reviewed.

3. THE SANDIA CATEGORIZATION FRAMEWORK

The Sandia Categorization Framework (SCF) is the computation engine used in the LSNAA for categorizing e-mail messages. It consists of several machine learning algorithms, data processing and transformation methods, and performance evaluation routines for solving two-class (binary) categorization problems. The LSNAA uses the SCF in the three tasks of categorizing e-mail messages into one of two classes:

1) *LSN Relevant* or *Not LSN Relevant*
2) *Federal Record* or *Not a Federal Record*, and
3) *Privileged* or *Not Privileged*.

To complete each of these tasks, a separate categorizer is built from (training) data that is manually labeled and then applied to a set of previously unseen (testing) data. A categorizer is a model of the characteristics of each document class computed using the terms in the training documents as features. Note that majority of work presented in this report was focused on the task of categorizing messages as either "LSN Relevant" or "Not LSN Relevant", although categorizers for each task listed above were created and tested.

Machine learning has been successfully applied to the problem of categorization of text documents across many document domains. However, there are several issues associated with these methods that we identify in this section and that anyone using such a system should understand. Machine learning methods and models are functions of the training data, and thus the performance of a categorizer depends on the reliability of the training data. Specifically, noise or contradictions appearing in the training data translate to ambiguity and poor performance in the categorizer models.

3.1. Determination of Methods to Include in the Framework

To determine the best methods for categorizing the e-mail, we first applied the standard machine learning methods available in the WEKA (Witten and Frank, 2005; WEKA, 2007) machine learning software library. This library contains a comprehensive set of the most current machine learning algorithms freely available and are is used to benchmark new and existing algorithms in new problem domains. The WEKA library is designed for algorithm and/or process pipeline design; therefore, many of the algorithms are not well suited for inclusion in production software. Thus, we used them only for prototyping algorithms for use in the SCF. The main drawbacks of WEKA to date include limited support for sparse data structures (which are crucial in text analysis where there may be little overlap in term usage for any given pair or collection of documents) and inefficient algorithm implementations (e.g., much of the code is intended for research and thus not analyzed and optimized for production). However, the use of WEKA provided a quick way to narrow down the set of algorithms to consider for the LSNAA.

3.1.1 Data

The original set of data available for training the categorizers consisted of 9429 e-mail messages labeled for each of the three binary classes by the originator of each message. As time was a

limited resource for this project, we chose to use this data set for investigating the utility of existing categorization algorithms even though we knew there were errors in the labels in the data.

The data was split into two mutually exclusive subsets of approximately equal size: one for building the categorizers and one for evaluating the performance of the categorizers. The subsets were randomly chosen from a uniform distribution and resulted in a training set of 4713 training 4716 testing documents. A STANLEY (Bauer *et al*., 2005) model was built for the training set using full documents (no splitting documents). The resulting model consisted of 15358 terms. Note that the training set consists of approximately 27.5 times as many negatives as positives with 4548 negatives and 165 positives.

3.1.2 Survey of Standard Methods

The methods surveyed in WEKA fall into one of the following categories: Bayesian, function, lazy, decision tree, decision rule, and miscellaneous. The Bayesian categorizers use Bayes' formula to model the probability that a given document belongs to a particular class. The lazy categorizers consist of methods for clustering the data and using the resulting clusters to determining the class label of a document: each cluster is associated with a document class and the cluster to which a testing document belongs determines its class label. The function categorizers use model regression fits to the features of the training documents in each class to determine a function to model the separation of classes. The decision tree and rule categorizers recursively partition the documents based on sequences of features either randomly chosen or chosen based on how well a feature can partition a subset of the documents into partitions containing documents of a single class. The resulting categorizers are either binary trees or a set of disjunctive rules that are used to separate documents into classes by following the logic represented in the tree or rule set. Finally, the set of miscellaneous categorizers consists of those that do not fit into any one of the other categories. Note that some of the methods are hybrid methods consisting of one or more methods combined in some specified way (such methods fall into the *meta-learner* category of learners in WEKA and are often referred to as *ensemble* methods).

The methods in WEKA surveyed for this work along with the performance results of each categorizer are presented in Appendix A. In Appendix A.1, we list the methods and the specific parameters used in building each categorizer; see WEKA (2007) for details on the WEKA methods. The results presented in this report are for those categorizers most commonly used in text categorization as well as those methods that performed the best in our initial investigations. At least 2 categorizers were included from each of the main types of methods available in WEKA.

Figure 3 presents one view of the results of the WEKA categorizers trained and tested on the data presented in the previous section; see Appendix A.2 for the complete set of results. In the figure, the number of *false negatives* is plotted against the number of *false positives* for each categorizer built to find e-mail messages labeled "LSN Relevant". Negative and positive labels are specified relative to the class of interest in building and applying a categorizer; thus in this case false negatives are those messages that are incorrectly labeled as negative ("Not LSN

Relevant") and false positives are those labeled incorrectly as positive ("LSN Relevant"). We see from the figure that there is a general trend in the trade-offs associated with the WEKA methods: false negatives are reduced at the cost of introducing more false positives. Thus, reducing the number of false negatives is equivalent to reducing the number of messages in which we are interested that we will miss (false negatives) while increasing the number of messages that will be labeled as those of interest (false positives). We address this trade-off more in Section 4, where we discuss the optimization of the categorizer used in the LSNAA.

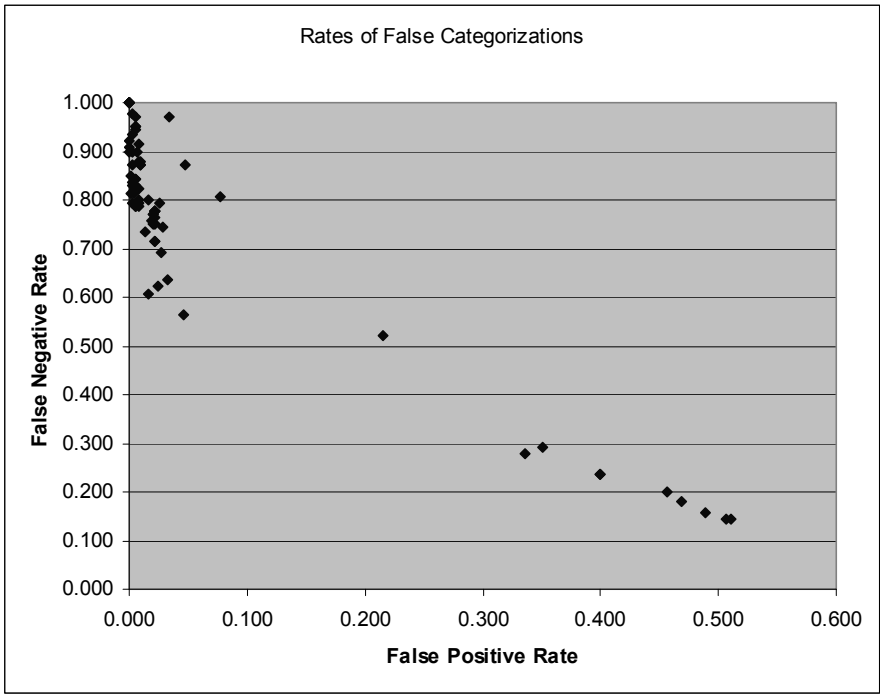

Figure 3. Rates of False Categorizations using the WEKA Categorizers.

3.2 Description of the Learning Methods

We present here the descriptions of the methods currently available in the SCF. The final set of categorizers chosen for the SCF includes several variants of those available in WEKA that performed the best on the data presented in the previous section and one method implemented by our group for another software library that was not available in WEKA.

The methods in the SCF can be trained on data labeled for two classes and the resulting categorizers generate two outputs: a categorization label and a confidence value. The label is either positive, negative, or is empty (corresponding to an undecided label). The confidence value is a relative measure of the confidence that the categorizer has in the corresponding label and is a function of the amount and characteristics of the training data along with the categorizer method parameters used to build the categorizer. Thus, confidence values have different meaning for each categorizer method.

3.2.1 Perceptron

For binary categorization problems where the data has a vector representation (such as the vector space model), linear binary categorizers are natural categorization functions to learn. Such categorization functions take the form of

$$f(\mathbf{x}) = b + \sum_{k=1}^{N} w_k \times x_k$$

with a threshold applied of $f(\mathbf{x}) > 0$. Here, $\mathbf{x}$ is the data input vector (N elements), $\mathbf{w}$ is the so-called weight vector (N elements), and b is a bias term. This function represents a decision space separated by the hyperplane $\mathbf{w}$ with offset b. The confidence value output by a linear binary classifier is the absolute value of $f(\mathbf{x})$. This means that inputs that are further away from the decision hyperplane will have a higher confidence value.

The Perceptron algorithm (Rosenblatt, 1958; Mitchell, 1997) is a well-known algorithm for learning a binary categorization function. Although the algorithm has well understood limitations (Minsky and Papert, 1969), such as requiring for convergence that the data be linearly separable, in practice the Perceptron algorithm can work well for document categorization. This is because the large number of features in the vector representation of documents makes it more likely that the documents will be linearly separable or nearly linearly separable (Joachims, 1997). In addition, the "kernel trick" can be applied Perceptron algorithm in order to handle data that is not linearly separable and even be applied to data that is not in a vector representation (Shaw-Taylor and Cristianini, 2004).

We make use of a variation of the Perceptron algorithm that includes the ability for the algorithm to enforce a predetermined margin on the data. Enforcing a margin will typically result in a better fit hyperplane and can be viewed as an easy way to obtain some of the beneficial separation properties of the computationally more expensive methods employing a Support Vector Machine (see Burges (1998) for a descriptions of those methods). Having separate parameters for the positive and negative margins allows the algorithm to place the decision boundary in different locations within the margin, which is useful in dealing with data where there is a large discrepancy between the number of positive and negative examples (which is the case for the LSN data).

3.2.2 Random Decision Tree

Tree-based methods (Breiman, *et al.,* 1984; Quinlan, 1993) partition the feature space of the input data and fit a simple categorization model within each partition. A decision tree is created by splitting the training data associated with each node according to a function of a single feature. At each leaf node of the tree, the training data is split using the feature which produces the best partitioning with respect to some error function. A restriction in most tree-based methods is that each feature can be used for splitting only once along each path from the root node to a leaf node. Construction of the decision tree is complete when each leaf node is a pure node, i.e., the node only contains data from a single category.

Random decision trees are decision trees that choose the feature on which to split the training data at each node from a random sample of the full set of features available. Thus, each node is split based on the best partitioning with respect to a random subset of features and not the entire feature set. This reduces the amount of computation required for building the decision tree, but in general introduces more error into the categorization model. For this reason, random decision trees are often used in ensembles (see Section 3.3.2 for more on random forests, which are ensembles of random decision trees).

The random decision tree method in the SCF is based on the CART (Breiman, *et al.,* 1984) decision tree method, a tree-based method which recursively partitions the feature space into binary partitions using the metric of *information gain* (Quinlan, 1986) for determining which feature (and the particular value of that feature) to split each node on. The random subsets of features are chosen from a uniform distribution of the available features at each set (i.e., those features not used to split any of its parent nodes).

One general drawback of decision trees used for categorization is that the resulting models tend to overfit the training data. That is, the models are highly specific to the training data and exhibit poor performance when used to categorize new data. To remedy this, tree-based methods often prune the resulting decision trees, a process of collapsing nodes based on some pruning criteria (see Breiman *et al.* (1984) for details.). Currently, the decision tree methods in the SCF do *not* employ pruning to address overfitting, as suggested in Banfield *et al.* (2007).
The confidence value output by decision trees in the SCF is either `-1`, `+1`, or `0`, corresponding to a negative, positive, or undecided label. Undecided labels occur often when there are too few features in the training data on which to split tree nodes. For large collections of textual data, this is often not an issue since the features are often functions of the terms occurring in the data.

3.2.3 Naïve Bayes

The naïve Bayes classifier is one of the most straightforward and easy to implement categorizers (Barber, 2005; Eibe and Bouckaert, 2006; Eyherandy, *et al.,* 2003; McCallum and Nigam, 1998; Rennie, 2001; Shen and Jiang, 2003). Naïve Bayes classifiers compute a class likelihood probability for each sample presented. The class likelihood probability is a product of the class prior and all feature value/class conditional probabilities estimated from the training data. (Actually for a specific sample, described by *n* features, for which the features take on the values 0 and 1 only, those feature/class conditional probabilities that match these values are used in the class likelihood estimates; for more details see Appendix E). The class with the maximum likelihood is returned as the class value for the sample presented. It will be most straightforward to describe the binary feature case (i.e., each feature is either present or not present with respect to a particular sample) here, so the following discussion will be restricted to the binary naïve Bayes classifier.

The binary naïve Bayes classifier relies upon strong independence assumptions, which can be wrong (i.e., naïve). In this classifier, each feature/class pair is assumed to be independent from every other feature/class pair. This assumption allows the probability of a feature obtaining a particular value given a specific class to be estimated very simply by the frequency of occurrence of that feature value over all training data at hand for the same class. None of the other features

or any of the other classes needs to be considered for this specific feature/class pair. Implicit in the assumption that all feature/class pairs are independent is the assumption that all classes are also independent. This second assumption allows class prior probabilities to be estimated directly from the frequency of occurrence of each class in the training data. Given these two quantities (class prior and feature/class conditional probabilities), we can uses Bayes theorem to estimate the class/feature conditional probability to within a constant (which is the feature probability). Note that the feature probability is constant across all classes (i.e., independent of class) and thus we can compare class/feature conditional probabilities to one another which cancels out the feature probabilities. The class likelihood is estimated by multiplying together each of the class/feature conditional probabilities for a specific class and specific feature values.

The naïve Bayes classifier has the advantage that it is easy to train and efficient to use in classification tasks. The class prior and feature/class conditional probabilities can be estimated in one pass through the training data. In the binary naïve Bayes classifier, feature/class conditionals are estimated for the existence of the feature (i.e., the feature is present, and thus the value is 1). The feature/class conditional for the absences of the feature is just 1 minus the feature/class conditional for the existence of the feature. The class likelihoods are computed as a product of the class/feature conditionals (i.e., for a particular sample, if the feature is present, use the feature/class conditional corresponding to the existence of the feature otherwise use thee feature/class conditional corresponding to the absence of the feature).

Because the naïve Bayes classifier uses products of conditional probabilities, one disadvantage is that a 0 probability will cause the entire product to be 0 as well. One way to address this problem is to ensure that each conditional probability is greater than or equal to 0 (usually, a small positive value is used). Because the values of the class likelihoods can be very near 0 or 1, it can be problematic to define a confidence value for the classification produced by the naïve Bayes classifier. Computing log-likelihood instead of raw likelihood can help somewhat in these cases, especially when there are many features (for more detail, see Appendix E). The confidence returned by the binary naïve Bayes classifier is the 1 divided by the quantity 1 plus the distance from 0 in log-likelihood space for the classification. Note that the logarithm of probabilities will always be less than or equal to 0, thus we use the negative of the logarithm for distance.

Another concern with the naïve Bayes classifier is that it can be too naïve in some cases, but given its learning and classification speed, it is a good candidate for inclusion in an ensemble learner.

3.2.4 Fuzzy ARTMAP

The Fuzzy ARTMAP neural network classifier operates by slicing the feature space up into rectangles (actually in the *n*–dimensional feature space these are hyper-rectangles), and then it assigns a single class (label) to each rectangle (Carpenter, *et al.*, 1992). During classification, a sample is given the class (label) of the closest matching rectangle. Note that these rectangles do not need to be independent of one-another, and in fact, two particular rectangles can partially overlap or one can be a proper subset of the other (however, it will never be the case that two different rectangles cover the exact same feature space). Each rectangle and all samples which are most closely matching to it (both inside the rectangle and immediately outside the rectangle)

are assigned the same (single) class (label) during classification. This section contains a brief algorithmic description of the Fuzzy ARTMAP neural network classifier (for a more detailed description, see Appendix F.

Even though the methodology of Fuzzy ARTMAP sounds simplistic it does support universal function approximation (Verzi, *et al.*, 2003; and Verzi, 2003) as do other neural network classifiers (Cybenko, 1989; Hartman, *et al.,* 1990; Hornik, *et al.*, 1990; Park and Sandberg, 1991). Classifiers that support universal function approximation have the capacity to represent many families of interesting functions. Most of the proofs supporting universal function approximation cover the family of continuous functions (or partially continuous functions with specific characteristics). The Fuzzy ART and Fuzzy ARTMAP neural networks support universal approximation of measurable functions (which is a slightly larger family than continuous functions).

The Fuzzy ARTMAP neural network classifier is composed of two Fuzzy ART neural networks (one each on the left and right) connected through a map field (Carpenter, *et al.,* 1991). Each Fuzzy ART neural network operates in an unsupervised fashion to form minimum sized rectangles containing all data that define the clusters (mathematically the rectangle is the minimum sized *n*–dimensional hyper-rectangle containing all training samples belonging to the cluster). The map field is used to ensure that each cluster on the left is connected (or linked or associated) with only one cluster on the right. Clusters on the left consist of rectangles covering a portion of the domain, and clusters on the right represent patterns of classes (labels). In most cases of classification with the Fuzzy ARTMAP, there will be a single right-side cluster for each class (label), and this will be true here (in SCF) as well. In this way, (fully) supervised learning can be achieved using Fuzzy ARTMAP, and classes will be partitioned independent from one another (meaning that each left-side cluster will be linked to a single class, and all samples belonging to this cluster will be given the same class label during classification).

Fuzzy ARTMAP is an on-line learner, which is different from many of other classification techniques (see Verzi (2003) for a more detailed description of the differences between on-line and off-line learners). As an on-line learner, Fuzzy ARTMAP considers only a single training instance at a time. An on-line learner does not know when it is done training, it only knows about the "current" training instance (and class). Even though it is an on-line learner, Fuzzy ARTMAP is trained using batch training (as are all of the other SCF classifiers). Thus, an external trainer will run Fuzzy ARTMAP through the entire training set several times until it stabilizes. Each pass through the training data is called an epoch, and stabilization is achieved when no weights are changed during an epoch of training. When Fuzzy ARTMAP is allowed to fully stabilize, it will create enough left-side rectangles so that each training sample is correctly classified (assuming that the training data is self-consistent). Binary-valued Fuzzy ARTMAP is guaranteed to stabilize in *n* epochs for *n*–dimensional feature space data (Georgiopoulos, *et al.*, 1994).

As an on-line learner, Fuzzy ARTMAP more closely models biological learning situations (Grossberg, 1980; Carpenter, *et al.*, 1992; Carpenter and Grossberg, 2003), but because of this, it can perform less well than other classifiers that are allowed to consider all of the training data and associated features including various statistical relationships. On-line learners are sensitive to the order of training data (even when they are trained with batch training). This means that

Fuzzy ARTMAP is a good candidate for inclusion in an ensemble (especially using different orderings of the training data).

In general, Fuzzy ARTMAP is fast to train and classify samples. However, with very large feature spaces (i.e., many feature dimensions), both the training and classification speed of Fuzzy ARTMAP can decrease significantly. Thus, Fuzzy ARTMAP is a good candidate for ensemble methods that use only a portion of the feature space for each of their constituents. A major advantage of Fuzzy ARTMAP is that it serves as a biologically inspired model of human (visual) category formation. In general, Fuzzy ARTMAP does not require any parameter tuning, but for optimal results some parameters might need to be adjusted.

A Fuzzy ARTMAP neural network classifier might over-fit the training data when it is allowed to fully stabilize (i.e., using batch training). However, Fuzzy ARTMAP can also be operated in fully on-line mode (e.g., trained on-line, too) in which case it never stops learning and it can under-fit the data or even lose track of previously learned classifications (when long-term memory is allowed to decay). In the SCF, we train Fuzzy ARTMAP using batch training either by itself or in ensembles.

The confidence value for a Fuzzy ARTMAP classification can be computed several different ways. For the SCF, confidence is calculated as

$$\frac{1}{1 + (\text{distance from cluster centroid})}.$$

Exact boundaries between classes can be very difficult to characterize in Fuzzy ARTMAP.

3.2.5 Scaled Categorizers

The confidence value output by the different methods presented in this section can differ in several orders of magnitude. This makes comparing or combining of the resulting categorizers problematic. To address this issue, a scaled categorizer is included in the SCF. A scaled categorizer is simply a categorizer whose confidence values are mapped to the range [`-1`, `+1`], with negative (positive) values corresponding to negative (positive) labels. An exact value of `0` corresponds to an undecided label. Since we are mostly concerned with binary categorization (i.e., the output classification is one of two labels) this scaling is reasonable. A value of 1 means that we are perfectly confident that the classification is correct, at a value of -1 we are perfectly confident that the classification is incorrect, that is, it should be the other class (of the two classes), and at a value of 0 we are not confident in either class as a correct classification. Note that at the boundary between the two classes, the scale would be 0. With multi-class categorization, we could use a scaling factor for each class or some reasonable approximation.

Perceptron categorizers are scaled by the maximum distance from linear separator over all the training data. Decision trees need no scaling as their confidence values are already mapped to [`-1`, `+1`]. Naïve Bayes categorizers produce confidence values that are already scaled between 0 and 1, thus this value need only be multiplied by the class value (either -1 or 1). And the output

from Fuzzy ARTMAP categorizers is also normalized between 0 and 1, thus it need only be scaled by the class value, being either -1 or 1 for the binary categorization in SCF.

3.3 Description of the Ensemble Learning Methods

We present here the ensemble methods currently available in the SCF. Categorization ensembles (also known as committees or meta-learners) are collections of different categorizers used to perform task. The output of the ensemble categorizer is a weighted combination of the individual categorizers. Categorization ensemble methods differ from one another in the individual categorizers used in the ensemble, how they use the training data to build each individual categorizer, and how the individual categorizers are weighted to produce a final categorization label. The results of the survey presented in Section 3.1.2 indicate that the majority of ensemble methods (i.e., meta-learners) performed well in labeling a high number of true positives (finding messages of interest) and a low number of false negatives (reducing the number of messages of interest incorrectly labeled). Thus, we included several ensemble methods in the SCF that performed well in the initial investigation as well as a new method developed to address the specific problem for which the LSNAA was developed.

3.3.1 Weighted Ensembles

The most basic ensemble method in the SCF consists of a collection of categorizers and is called a *weighted ensemble method*. Associated with each categorizer is a weight for scaling its confidence output score. For each message to be labeled, the ensemble output is a sum of the outputs of each individual categorizer, scaled by its corresponding weight.

The default weighting scheme is $1/N$ for an ensemble of N categorizers, giving equal weight to each categorizer. Equal weighting ensembles can be biased by categorizers with possible confidence values that are orders of magnitude greater than those possible for all other categorizers in the ensemble. Thus, scaled categorizers should be used with equal weighting ensembles. When scaled categorizers and equal weighting is used, the resulting categorizer is equivalent to the standard majority voting ensemble method (Sebastiani, 2002).

Another weighting scheme implemented uses training set error performance to weight the ensemble members. The goal of this weighting scheme is to incorporate more of the characteristics of the categorizers with fewer errors into the final ensemble categorizer.

The categorizer used in the LSNAA is a weighted ensemble of several different categorizers, including ensemble categorizers. See Section 4 for more details on the particular ensemble used, the weights associated with each individual categorizer, and the rationale behind these choices.

3.3.2 Random Forests

The *random forest ensemble method* (Breiman, 2001) in the SCF produces an ensemble of random decision trees with equal weighting. The number of individual random tree categorizers and the number of random features to use in training each individual categorizer can be

specified. The random forests categorizers created in our study of the WEKA categorizers performed well in labeling a high percentage of true positives and relatively few false negatives. Other research groups have also found that random forest categorizers perform well for the related problem of categorization of spam in e-mail messages (Koprinska, 2007).

3.3.3 Balanced Learners

A new ensemble method developed for solving the problem for which the LSNAA was developed is called a *balanced learner ensemble method*. This method was developed to address the highly skewed distribution of classes in the e-mail data. Recall that the ratio of negative to positive instances in the training data used for the WEKA study was nearly 30 to 1 and thus presented a challenge for categorizers to label all of the positives correctly.

The balanced learner ensemble method is based on the ensemble method presented in (Gao *et al.*, 2007). In that work, the class with more instances is randomly partitioned such that each partition is approximately equal in size to the class with fewer instances in the training data. The result is a collection of training sets where each set contains all of the instances of the minority class and a random sample (with no duplicates across the entire set) of those of the majority class. That method is available in the SCF but not implemented in the LSNAA. Our balanced learner differs from the work of Gao, *et al.* as follows: 1) the majority class is sampled with replacement; 2) skewed distributions in the classes in the balanced training sets are allowed (e.g., in order to approximate the original distribution but with less skew); and 3) more than $k = \lceil (N - n)/n \rceil$ individual categorizers are allowed in the ensemble, where N is the total number of training instances and n is the number of training instances in the minority class.

The number of individual categorizers, the number of random samples taken from the instances of the majority class for the training set of each individual categorizer (the default is the number of instances in the minority class), and the weight associated with each individual categorizer (the default is an equal weighting scheme) can be set for each balanced learner ensemble categorizer.

In all of our testing, the balanced learner ensembles of categorizers from the SCF improved the performance of those individual categorizers. We believe that this performance gain is problem dependent and a result of the highly skewed class distribution of the data for this problem. There is much more research to be done in understanding the extent of problems for which this method is most applicable and how best to choose the method parameters for particular problems and data.

4. USING THE SCF FOR CATEGORIZING LSN E-MAIL MESSAGES

The development and evaluation of several SCF individual categorizers and ensemble categorizers applied to two collections of e-mail data is discussed in this section. Also, the data used for the performance evaluations, the strategies for tuning and optimizing the categorizers, and the results of the evaluation are described. Specifically, the data was divided into two collections: Case 1 (messages sent before October 2006) and Case 2 (messages sent after October 2006).

There are many performance metrics used by researchers in the machine learning community for evaluating categorizer methods; see Sebastiani (2002) for a recent survey. In this report, we present the raw output (in Appendix B) in terms of positive categorizations ("LSN Relevant", "Federal Record", "Privileged") and negative categorizations ("Not LSN Relevant", "Not a Federal Record", "Not Privileged") correctly and incorrectly labeled along with accuracy, precision, and recall, precision-recall curves (used originally for evaluating performance of information retrieval systems but now often used for categorizer evaluation), and a metric specifically design to highlight performance with respect to the goals of the LSNAA (relatively high number of true positives and low number of false negatives simultaneously). These are the tools we used in determining the "best" categorizers to be used in the LSNAA.

As each categorizer method contains user-specified several parameters, performance of the resulting categorizers are directly dependent on the choices for these parameters. However, it is often not clear what the best parameter choices are for a given training set, and the performance of the categorizer on the testing data may differ from that of the performance on the training data for a given set of parameter values. Thus, tuning and optimization of the categorizers must be performed on a given training data set prior to use. The SCF categorizer used in the LSNAA is an ensemble categorizer, and the choice of weighting for this ensemble is presented in this section. Validation of the categorizer used in the LSNAA is also presented.

The solution process described in Section 1.4 includes iterate over the steps of building, testing, and applying categorizers to data until an acceptable categorizer has been determined. For the problem and data which motivated the development of the LSNAA, it was determined that two iterations of these steps were adequate (as assessed by SMEs associated with the problem and data). The following sections describe these two iterations in detail.

4.1 First Review Categorization and Performance Evaluation

4.1.1 Validation Data

A subset of 1766 e-mail messages from those used for the WEKA study and described in Section 3.1.1 was chosen for optimizing and evaluating the performance of the SCF categorizers. These messages were then certified by a group of SMEs trained on the process and procedures for labeling the messages with respect to the three categorizations listed at the beginning of Section 3. Each of the messages was certified by at least two experts.

The numbers of messages for each of the three types of labels are as follows:

- Number of *LSN Relevant* documents: 83 (5%)
- Number of *Federal Record* documents: 1564 (89%)
- Number of *Privileged* documents: 291 (16%)

Thus, the class distribution for each type is highly skewed, but in different directions for the "LSN Relevant"/"Privileged" messages and those labeled as a "Federal Record". Note that this is automatically adjusted for in the balanced learner ensemble methods, since for each categorizer the minority and majority classes are first identified and then the training data samples are chosen.

Note that only e-mail messages from Case 2 were validated by SMEs in the first review, and thus are the only messages used in training and assessing performance of the categorizers.

4.1.2 Parameter Identification for Individual Categorizers in the LSNAA

Categorizer-specific parameters for individual categorizers were identified by starting with defaults values discussed in the machine learning community and then tuned by means of a simple local search in parameter space around those initial values. This process helped us assess the robustness of these methods and defaults values (by identifying correlated parameters and approximating sensitivity of the parameters near the default values). We emphasize that this process was not a rigorous numerical optimization of the categorizer parameters for the given training data. Such a process would require more time and effort than was available and would not necessarily produce an optimal categorizer for all testing data to be labeled since the relationship between the training and testing data is unknown.

Figure 4 presents the precision-recall curves of several of the categorizers used in the LSNAA. *Precision* is defined as the percentage of true positives in the testing set that were categorized correctly:

$$precision = \frac{TP}{TP + FP}$$

and *recall* is defined as the percentage of the testing set categorized as positives that were true positives:

$$recall = \frac{TP}{TP + FN}$$

where TP denotes true positives (positive data categorized as positive), FP denotes false positives (positive data categorized as negative), and FN denotes false negatives (negative data categorized as positive). Precision-recall curve plots are typically used to compare categorizers with respect to the tradeoff of precision and recall. With many categorizers, however, it is difficult to compare performance, as can be seen in Figure 4.

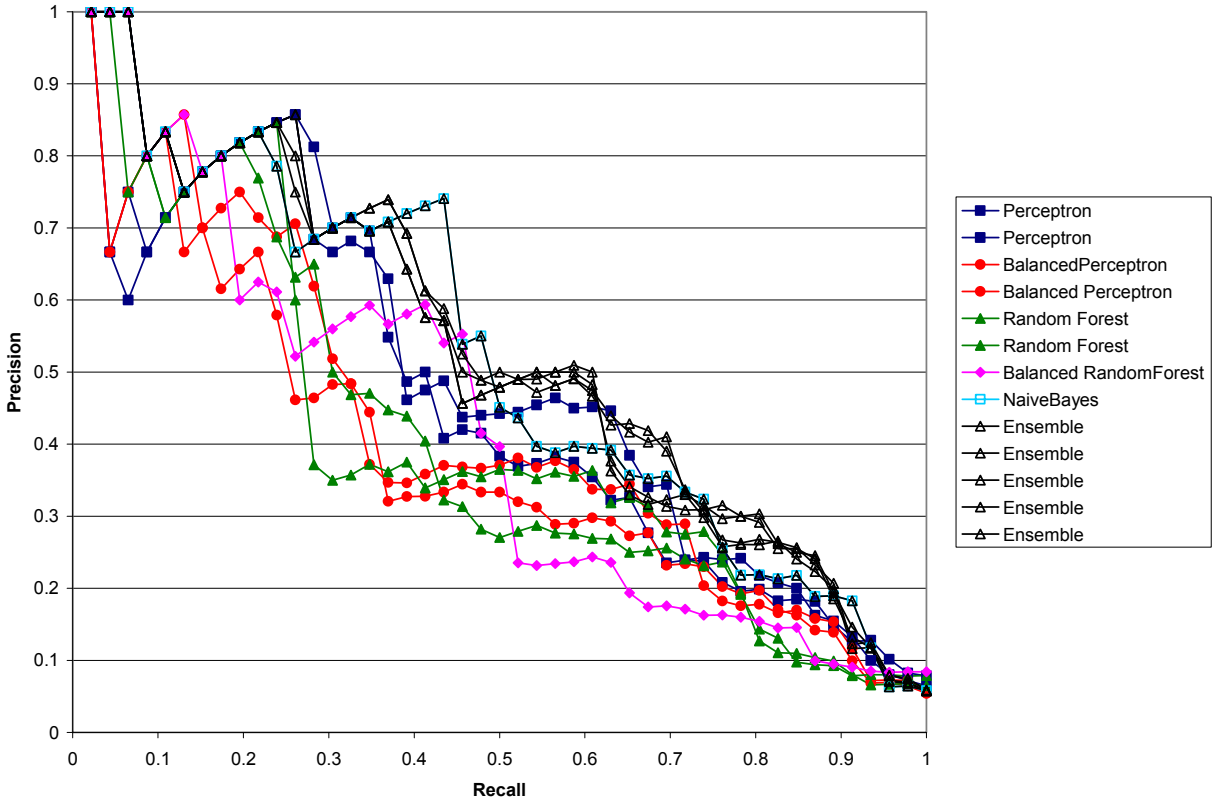

Figure 4. Precision-Recall Curves of Several SCF Categorizers.

We used the precision-recall curves of the several different categorizers parameterized in several ways to determine the set of categorizers to be used in the ensemble categorizer for the LSNAA. Appendix D.1 presents the results of the parameter tuning for the perceptron, balanced perceptron, random forest, balanced random forest, naïve Bayes, and balanced naïve Bayes categorizers. Below are the categorizers that were chosen for the initial review based on our desire to create an ensemble categorizer with both high recall and high precision, with high recall taking precedence:

- **Perceptron Categorizer**
  Note: Many iterations used for greater accuracy, higher margin of positive instances results in a categorizer which labels positive that are clearly different than the negatives.
- *Parameters:*
  - `Maximum iterations = 5000`
  - `Margin of positive instances = 100`
  - `Margin of negative instances = 1`

- **Balanced Perceptron Categorizer**
  Note: Fewer iterations and smaller margins between classes result in less accurate individual categorizers but the balancing of training data uses these weak approximations to create a more robust categorizer for determining true positives.
  Parameters:
  - `Number of individual categorizers = 10`

    - Majority instances = 2.5 (times the number of minority instances)
    - Maximum iterations = 100
    - Margin of positive instances = 50
    - Margin of negative instances = 10

- **Balanced Random Forest Categorizer 1**
  Note: A large ensemble of random decision trees trained on slightly skewed training data samples removes a lot of the noise in the features characterizing the positives and negatives.
  Parameters:
    - Number of individual categorizers = 200
    - Majority instances = 1.5 (times the number of minority instances)
    - Number of random features used per tree node = 100

- **Balanced Random Forest Categorizer 2**
  Note: The fewer ensemble members trained on data that better reflects the skewed class distribution of the data results in a more robust categorizer when the training and testing data is very different.
  Parameters:
    - Number of individual categorizers = 10
    - Majority instances = 2.5 (times the number of minority instances)
    - Number of random features used per tree node = 100

- **Naïve Bayes Categorizer**
  Parameters:
    - Minimum feature probability = 0.00001

4.1.3 Weight Identification for the Ensemble Categorizer in the LSNAA

An ensemble of categorizers was chosen for the LSNAA based on the performance of the individual categorizers presented in the previous section. We chose to include categorizers that performed well in either labeling a high number of true positives or a low number of false negatives, or both. In this way, categorizers would be included in the final ensemble that could collectively perform well with respect to the given tasks, and the contribution of each individual categorizer could be controlled using the ensemble weights. The particular weights for the ensemble members were chosen using the same process for individual categorizer parameter identification discussed in the previous section; specifically, an ensemble categorizer with high recall and high precision was determined to be the best categorizer.

Appendix B.2 presents a sample of the results of the simple weighting local search around the equal weights of `1` for each ensemble member except for the naïve Bayes categorizer. Due to the amount of time required to build and test the naïve Bayes categorizer, the weight was determined once the other weights had been chosen.

Figure 4 also shows the precision-recall curves of several of the ensembles using different weighting schemes. Note that the ensemble categorizers in general perform as well or better than the individual categorizers, as is expected.

The weights used for the categorizer in the LSNAA for the first review were as follows:

- `Perceptron = 2.5`
- `BalancedPerceptron = 1.5`
- `BalancedRandomForest1= 5.0`
- `BalancedRandomForest2= 2.5`
- `NaiveBayes = 2.5`

4.1.4 Performance Assessment

The validation of the SCF ensemble categorizer used in the LSNAA was performed in two steps. The first step consisted of 10-fold cross validation, and the second of a 100-fold random validation.

Table 1. Results of 10-Fold Cross Validation for the Ensemble Categorizer in the LSNAA.

LSNRelevant : Training							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	39.960	762.970	91.030	5.040	0.893	0.311	0.888
STD	2.291	20.689	20.689	2.291	0.022	0.040	0.051
LSNRelevant : Testing							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	31.230	733.590	110.490	10.790	0.863	0.225	0.745
STD	4.381	28.552	25.162	3.862	0.026	0.039	0.079
Privileged : Training							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	130.500	689.070	69.930	9.500	0.912	0.653	0.932
STD	2.684	10.460	10.460	2.684	0.012	0.036	0.019
Privileged : Testing							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	123.630	638.240	102.810	21.420	0.860	0.548	0.853
STD	7.534	20.302	14.649	4.473	0.015	0.040	0.028
FederalRecord : Training							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	800.960	58.920	36.080	3.040	0.956	0.957	0.996
STD	1.608	6.297	6.297	1.608	0.007	0.007	0.002
FederalRecord : Testing							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	778.160	27.230	74.550	6.160	0.909	0.913	0.992
STD	18.679	4.627	8.543	2.557	0.009	0.009	0.003

Cross validation is the process of splitting the validated data into equal-size, randomly chosen subsets (folds), and testing on each subset while training on the combination of the other subsets. Thus, in 10-fold cross validation, the validated data is split into 10 subsets, and there are ten sets of results, which are averaged to evaluate the overall performance. This process of validation is an attempt to remove the bias of a particular split used in training and testing the data. Table 1 presents the average and standard deviation of the cross validation results across 10 folds for the ensemble described in the previous section.

Random validation is similar to cross validation except that the folds consist of approximately equally-sized random splits of the data into training and testing data. Thus, in 100-fold random validation, the validated data is randomly split 100 times, and there are 100 sets of results, which are averaged to evaluate the overall performance. This process of validation is also an attempt to remove the bias of a particular split used in training and testing the data, but allows for replacement of the training data. Table 2 presents the average and standard deviation of the random fold validation results across 100 folds for the ensemble described in the previous section.

Table 2. Results of 100-Fold Random Validation for the Ensemble Categorizer in the LSNAA.

LSNRelevant : Training							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	67.500	1414.900	106.100	1.500	0.932	0.389	0.978
STD	1.269	5.065	5.065	1.269	0.003	0.011	0.018
LSNRelevant : Testing							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	6.500	153.300	15.000	1.800	0.905	0.315	0.802
STD	1.958	4.001	4.830	1.229	0.024	0.107	0.130
Privileged : Training							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	254.300	1294.100	36.900	4.700	0.974	0.874	0.982
STD	2.869	5.877	5.877	2.869	0.005	0.018	0.011
Privileged : Testing							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	24.100	133.500	14.000	5.000	0.892	0.633	0.831
STD	2.998	3.923	2.867	2.449	0.024	0.067	0.074
FederalRecord : Training							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	1404.400	126.100	57.900	1.600	0.963	0.960	0.999
STD	1.350	3.510	3.510	1.350	0.003	0.002	0.001
FederalRecord : Testing							
	TP	***TN***	***FP***	***FN***	***Accuracy***	***Precision***	***Recall***
Average	154.700	6.200	14.000	1.700	0.911	0.917	0.989
STD	5.165	2.201	4.899	1.252	0.025	0.029	0.008

4.2 Second Review Categorization and Performance Evaluation

The final ensemble categorizer from the first review was applied to the two collections of e-mail messages, and then the SMEs validated a new subset from both of the Case 1 and Case 2 collections.

4.2.1 Validation Data

A subset of 34151 e-mail messages was chosen for optimizing and evaluating the performance of the SCF categorizers in the second review. These messages were again certified by a group of SMEs trained on the process and procedures for labeling the messages with respect to the three categorizations listed at the beginning of Section 3. Each of the messages was certified by at least two experts.

The numbers of messages for each of the three types of labels are as follows:

- Number of *LSN Relevant* documents: 7582 (22%)
- Number of *Federal Record* documents: 32891 (96%)
- Number of *Privileged* documents: 3140 (9%)

Note the class distribution for each type is skewed as in the first review. However, the amount of skew is different. Specifically, there is significantly less skew in the distribution of "LSN Relevant" and "Not LSN Relevant" messages. These differences highlight the need for iterating through the steps of the solution process of determining an effective categorizer. As the set of validated data changes, the categorizers should be updated to reflect this new information.

4.2.2 Parameter Identification for Individual Categorizers in the LSNAA

The same process for identifying effective parameters for the individual categorizers was then applied to the validation data. The only difference was that the set of parameters searched were not centered at the defaults suggested by the algorithm developers. The parameters were centered around those determined to be the best parameters during the first review.

Appendix D.2 presents the results of the parameter identification studies for the second review. Note that the individual categorizers used were slightly different than those during the parameter identification study of the first review; they reflect the set of categorizers used in the final ensemble categorizer of the first review.

Figure 5 presents a plot of the false negatives by the false positives for the categorizers created during the parameter identification study. This type of plot is useful in determining rough bounds on the performance of the individual categorizers being used. For example, Figure 5 illustrates the trend of the categorizers and a (disconnected) lower bound of the tradeoffs between the two types of false categorizations that should be expected using the set of parameters identified.

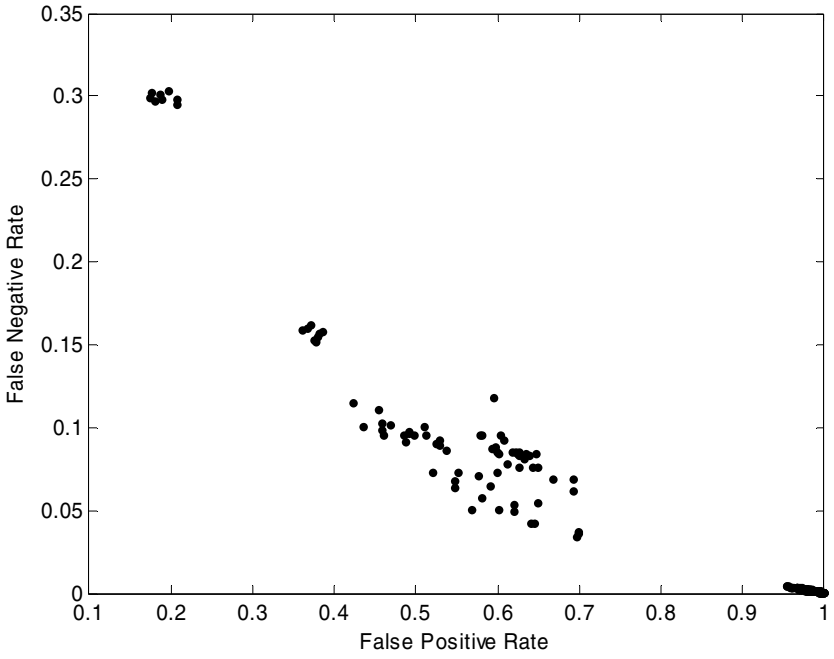

Figure 5. False Categorizations of the Individual Categorizers: Second Review, Test Data.

The individual categorizer parameters chosen for the second review are as follows:

- **Perceptron Categorizer**
  Note: Many iterations used for greater accuracy, higher margin of positive instances results in a categorizer which labels positive that are clearly different than the negatives.
- *Parameters:*
  - `Maximum iterations = 3000`
  - `Margin of positive instances = 150`
  - `Margin of negative instances = 1`

- **Balanced Perceptron Categorizer**
  Note: Fewer iterations and smaller margins between classes result in less accurate individual categorizers but the balancing of training data uses these weak approximations to create a more robust categorizer for determining true positives.
  Parameters:
  - `Number of individual categorizers = 5`
  - `Majority instances = 2.0 (times the number of minority instances)`
  - `Maximum iterations = 500`
  - `Margin of positive instances = 1000`
  - `Margin of negative instances = 5`

- **Balanced Random Forest Categorizer 1**
  Note: A large ensemble of random decision trees trained on slightly skewed training data samples removes a lot of the noise in the features characterizing the positives and negatives.
  Parameters:
    - `Number of individual categorizers = 100`
    - `Majority instances = 1.0 (times the number of minority instances)`
    - `Number of random features used per tree node = 200`

- **Balanced Random Forest Categorizer 2**
  Note: The fewer ensemble members trained on data that better reflects the skewed class distribution of the data results in a more robust categorizer when the training and testing data is very different.
  Parameters:
    - `Number of individual categorizers = 10`
    - `Majority instances = 3.0 (times the number of minority instances)`
    - `Number of random features used per tree node = 50`

- **Naïve Bayes Categorizer**
  Parameters:
    - `Minimum feature probability = 0.00001`

4.2.3 Weight Identification for the Ensemble Categorizer in the LSNAA

The SCF code base was refactored between the first and second reviews to facilitate more efficient identification of the ensemble weights given the individual categorizers. Specifically, each individual categorizer was built and tested *once*, and then different combinations of ensemble weights were tested. This allows for a more thorough search of potential ensemble weights. Recall that although it is possible to perform a rigorous global optimization of the ensemble weights, this is not advised due to problems with overfitting of the testing data. Unless there is a guarantee that there will be no differences between the testing data used to create a categorizer and the data to which the categorizer will eventually be applied (i.e., statistically significant differences in the attributes of the e-mail messages), global optimization is not recommended.

For this problem, an exhaustive search of all combinations of ensemble weights in the interval of [0,5] in steps of 0.25 was performed. A plot of the false negatives by the false positives is shown in Figure 6. Note that the general trend of the individual categorizers appearing in Figure 5 also appears here in Figure 6 for the ensemble categorizers.

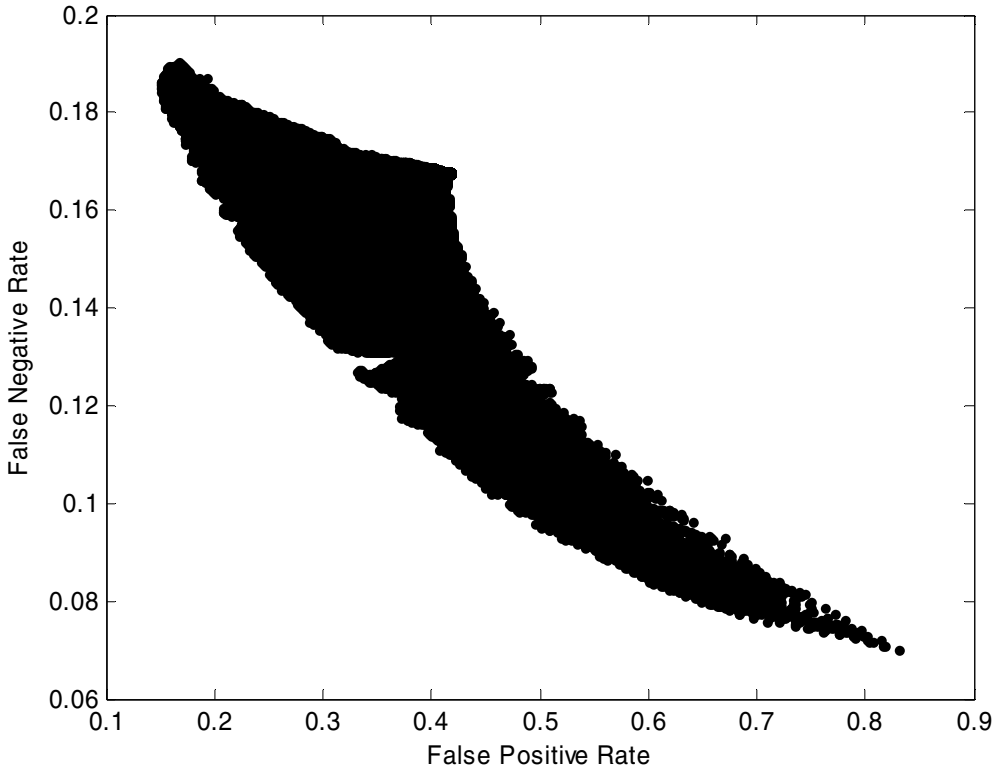

Figure 6. False Categorizations of the Ensemble Categorizers: Second Review, Test Data.

Since the search for weights was performed on a discrete grid, further investigation into the choice of ensemble weights was performed around those set of ensemble weights with the best performance. The weights used for the categorizer in the LSNAA for the second review were as follows:

- `Perceptron = 2.0`
- `BalancedPerceptron = 4.0`
- `BalancedRandomForest1= 3.0`
- `BalancedRandomForest2= 1.25`
- `NaiveBayes = -1.0`

These weights were chosen to reduce the false positive rate as the primary goal and the false negative rate as the secondary goal. Note that the ensemble weight for the naïve Bayes categorizer was not part of the initial study and was only determined by further investigation. The negative weight for the naïve Bayes categorizer was unexpected, as this indicates the use of the categorizer in the opposite way it was trained. We suspect that this behavior acts to balance the other categorizers in the ensemble. However, testing on more general sets of data would need to be performed to verify this claim.

4.2.4 Performance Assessment

Validation of the ensemble categorizer used in the second review was performed in the same manner as in the first review, and comparable results were found.

To assess the performance beyond the validation tests, the SMEs were presented with the results of the categorizations for the set of validated data. Figures 7 and 8 present the results of the categorizations for the Case 1 and Case 2 collections, respectively, as they were presented to the SMEs. The top plot in each figure shows a histogram of all the categorizations for the validated data. The bottom two plots in the figures show histograms of the false positives and false negatives for each validated data set. Note that in both figures the false categorizations are clustered around zero (0), indicating that the ensemble categorizer has no clear indication of the correct categorization. As the goal of LSNAA is to generate a prioritized list of likely candidates for a particular categorizations (as opposed to a list of categorizations that will be used for decision making directly), these results are very promising. The outcome if that SMEs or users need focus most of their effort on where the categorizer is most confused (near 0). This result is to be expected as a system categorizer *should* be identifying those parts of the data that are the most ambiguous in terms of categorization, which is what is happening in this case with the false categorizations clustered around zero.

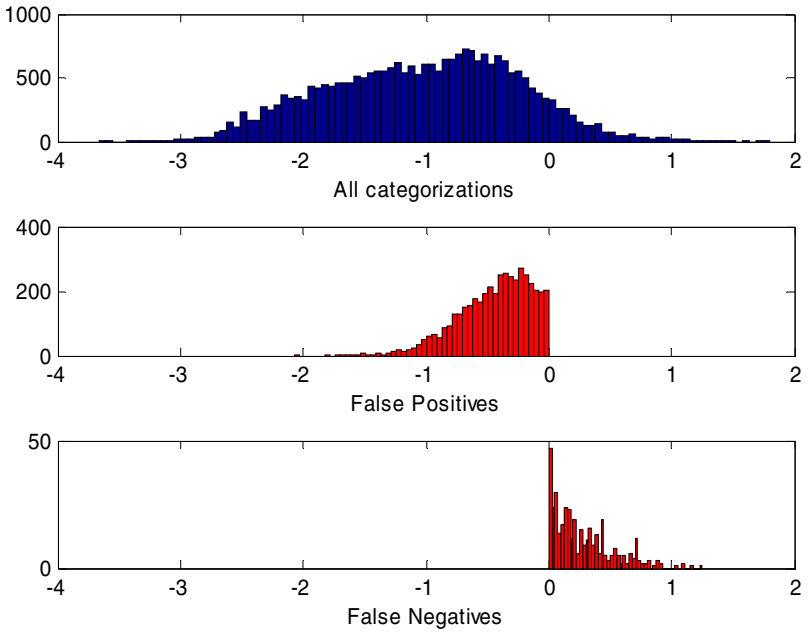

Figure 7. False Categorizations of the Ensemble Categorizers: Second Review, Case 1 Data.

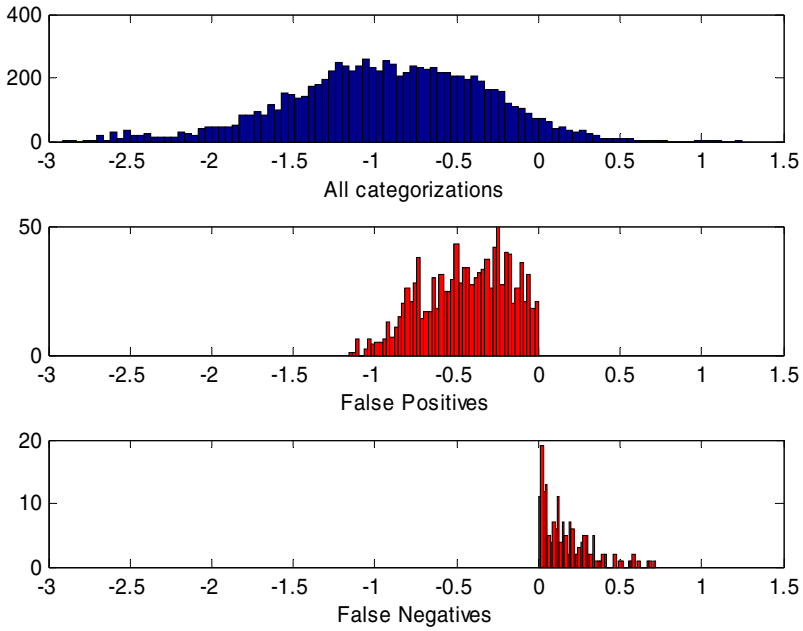

Figure 8. False Categorizations of the Ensemble Categorizers: Second Review, Case 2 Data.

5. FUTURE DIRECTIONS

In this section, ideas that have been identified for exploration in future versions of the LSN Assistant set of tools are presented.

5.1. Data Processing

E-mail messages often lack formal grammatical structure, contain sentences that are not fully formed, and contain many spelling errors. Such document characteristics present challenges for analyzing collections of documents. Below are several ideas that have shown promise for several text analysis projects in the machine learning research community.

5.1.1 Latent Semantic Analysis

Deerwester, *et al.* (1990) developed Latent Semantic Analysis (LSA) to help improve information retrieval systems by statistically correlating terms and documents, removing "noise", and reducing the dimension of features used to characterize each document of collections of unstructured text documents. We performed a preliminary investigation into using LSA to create the vector version of the document for the LSNAA; however, we found that it tended to reduce categorization performance at the various dimensions tested. One challenge in employing LSA is choosing how much feature dimension reduction and/or noise reduction should be performed for effective analysis (e.g. categorization for our problem). Choices are often based on heuristics and can be dramatically different for various data sets and algorithms. Thus, work on developing recommendations for the task of e-mail categorization or more generally, for automatically tuning the parameters for a given problem and data set should be explored to facilitate use of LSA in automatic categorizations systems. It may also be worthwhile investigating using the vectors generated from LSA as additional features rather than as a reduced set of features for this problem.

5.1.2 Natural Language Processing

Currently, the features used to characterize documents for use with the SCF categorizers are based on terms appearing in the documents. More sophisticated statistical natural language processing (NLP) techniques exist (Manning and Schütz, 1999) for extracting richer sets of features from documents than simply the list of terms appearing in a document. Specifically, several methods for noun phrase (i.e., named entity) extraction have been developed, and the extracted phrases could be used as more discriminating document features than individual terms alone. Along the same lines, language modeling using *n*-grams (sequences or windows of *n* consecutive terms) may also help to determine richer feature sets for use in creating categorizers. Other NLP techniques that should be explored include noun and verb phrase detection (i.e. chunking) and part of speech detection.

5.1.3 Feature Extraction and Selection

There are several other existing methods for generating features for documents than those currently used in the SCF (via STANLEY). Other methods that have been used in categorization

research include term extraction (e.g., LSA or term clustering) and term selection (e.g., based on document frequency, random selection, information theoretic metrics, etc.). The goal of these methods is to produce a reduced feature set to facilitate both computational efficiency and identification of the most important discriminating features of individual documents. One downside to most of these approaches is that the reduced space does not preserve the sparsity patterns of the original document term features. It is not clear what the impact of discarding such information has on the overall performance of categorizers. Thus, such methods should be studies more in the context of e-mail categorization particularly. See Sebastiani (2002) for more details of such approaches.

5.2. Categorization

Further work can also be done on the categorization algorithms themselves, particularly in utilizing more of the data through semi-supervised learning and automating more of the development process by automatically tuning categorization parameters and ensemble weights.

5.2.1 Semi-Supervised Learning

Algorithms for creating categorizers typically benefit from having access to large sets of training data. That is, the more data a categorizer is trained on, the more likely it will correctly categorize a typical document example. However, the creation and maintenance of large sets of training data is constrained by the time availability of SMEs associated with the problem and data of interest. Methods of *semi-supervised learning* (Chapelle, *et al.*, 2006) overcome this constraint by combining relatively small sets of validated training data and statistical techniques to increase the size of data used to train categorizers. This approach has shown promise for general data categorization, and may prove helpful for the problem of e-mail categorization.

5.2.2 Automatic Categorization Parameter and Ensemble Weight Tuning

One of the most time consuming steps in the current solution process from Section 1.4 is the building and testing of categorizers. Specifically, the tuning of the individual categorizer parameters and the weights used in the ensemble categorizer are labor-intensive, manual processes. Although some automation has been developed for creating and a large number of categorizers and sets of ensemble weights, the current approach requires detailed knowledge of the categorizers. Automated methods of determining such parameters and weights for a given problem and/or data set include ridge regression, lasso regression, and global direct search (Hastie, *et al.*, 2001). Such methods would allow for generation of more optimal categorizers and could generate useful statistics for analyzing the performance of the categorizers (including robustness/sensitivity and feasibility statistics).

Another potential benefit of incorporating automatic parameter and weight tuning into the SCF is support for performance tuning based on a complicated set of metrics. For example, in the application presented in this report, the goal was to generate categorizers biased toward the highest number of true positives and the least number of false negatives. By incorporating automatic methods for parameter and weight tuning, it should be straightforward to handle such metrics.

5.3 The LSN Real-Time Assistant

Although this report focuses on the LSNAA, the SCF and most of the other framework for processing data, testing categorizers, and applying categorizers to e-mail messages will be used in the LSN Real-Time Assistant (LSNRA) tool. However, the process of data processing, and categorizer creation and testing for the LSNRA includes a different set of requirements. Specifically, the issues of automation of all solution steps, efficient data and categorizer updating, and combining categorization rules with categorizations must all be addressed.

Scheduling of updates during system low-usage times, combined with online learning algorithms should facilitate efficient data and categorizer updates. Online learning algorithms create a categorizer in a single pass of the data, and thus facilitate addition of new (or updated) data more readily than batch learning algorithms, which must have access to all of the training data in order to build a categorizer. Currently, with the exception of the Perceptron categorizer, the SCF is comprised of batch learning methods. Those batch methods would have to load all existing data each time the training data is update and a categorizer is built, whereas the online learning methods need only update the existing categorizer using any previously unused training data.

Using a set of pre-defined categorization rules may help improve the performance of the SCF categorizers. Specifically, rules based on specific keywords (e.g., "DRAFT" "Calendar Request", "Training Requirement", etc.) may help identify specific categorizations of automatically generated e-mail messages or messages that are use only according to a particular institutional policy.

This page intentionally left blank.

6. CONCLUSIONS

The Licensing Support Network (LSN) Assistant is a system for categorizing e-mail messages and documents and investigating and correcting existing document archives. The two main tools in the LSN Assistant, the LSN Archive Assistant (LSNAA) tool for categorizing existing e-mail and documents and the LSN Real-time Assistant (LSNRA) tool for categorizing, were introduced, with the details of the LSNAA provided. The Sandia Categorization Framework (SCF), the categorizer engine behind the LSN Assistant tools was described in detail as well.

The design and implementation of the LSNAA database, system architecture, and graphical user interfaced (GUI) were described. This process was aimed at both satisfying the system requirements (Appendix A) and the addressing the suggestions of the SMEs and customers as the system tools were developed and deployed. Note that the LSN Assistant tools should be regarded as works in progress, with development active on the system architecture, the SCF, and the LSN Assistant application GUIs.

The problem of categorization of text documents in general and e-mail messages in particular is an active area of research, and the processes and results presented in this report were specifically designed and tuned for the particular problem and data associated with LSN e-mail messages. As noted throughout this report, the particular instances of categorizers used in the LSNAA are dependent on the training data, and thus are not useful for solving general text categorization problems. However, the steps and recommendations for processing data and creating, tuning, and testing categorizers presented in this report *are* applicable to solving text categorization problems in general. The specific examples and categorizers described in this report should serve to demonstrate choices made along the entire process and the implications of those choices with respect to performance and output of a categorizer.

As in most research associated with machine learning and the problem of data categorization, quantifying the confidence in the categorizers created is crucial in determining the general usefulness and applicability of the categorizers. The importance of the parameter and ensemble weight tuning, along with the categorizer validation (Section 4) presented in this report should not be underestimated. The use of categorizers for decision making is problematic due to the dependence on the particular data used for training. We have shown that using the methods presented in this report, a prioritization of categorizations and identification of regions of categorization ambiguity (Section 4.2.4) can be used to assist in challenging and complex large-scale decision making processes.

This page intentionally left blank.

7. REFERENCES

Banfield, Robert E., Hall, Lawrence O., Bowyer, Kevin W. and Kegelmeyer, W. Philip (2007). A comparison of decision tree ensemble creation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1), 173–180.

Barber, David (2005). *Learning from data 1: Naïve Bayes*, Retrieved October 13, 2005, Learning from Data: http://www.inf.ed.ac.uk/teaching/courses/lfd/lectures/lfd_2005_naive.pdf .

Basilico, Justin and Verzi, Steven (2006). STACY licensing support network assistant: Software design document, *Technical report,* Sandia National Laboratories..

Bauer, Travis, Laham, Darrell, Benz, Zachary, Dooley, Scott, Kimmel, Joseph and Oberbrekling, Rob (2005). Text analysis and dimensionality reduction for automatically populating a cognitive model framework. In *Cognitive Systems : Human Cognitive Models in Systems Design*, Lawrence Erlbaum, pp. 153–176.

Breiman, Leo (2001). Random forests. *Machine Learning*, 45(1), 5–32.

Breiman, Leo and Friedman, Jerome, Olshen, R.A. and Stone, Charles J. (1984). *Classification and Regression Trees*, Wadsworth.

Burges, Christopher J. C. (1998). A Tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2, 121–167.

Carpenter, Gail, Grossberg, Stephen and Rosen, David (1991). Fuzzy ART: Fast stable learning and categorization of analog patterns by an adaptive resonance system. *Neural Networks*, 4, 759–771.

Carpenter, Gail, Grossberg, Stephen, Markuzon, Natalya , Reynolds, John and Rosen, David (1992). Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions on Neural Networks*, 3(5), 698–713.

Carpenter, Gail and Grossberg, Stephen (2003). Adaptive resonance theory. In M. A. Arbib, (editor), *The Handbook of Brain Theory and Neural Networks*, MIT Press, pp. 87–100.

Chapelle, Oliver, Schölkopf, Bernhard and Zien, Alexander (Eds.) (2006). *Semi-Supervised Learning*. MIT Press.

Cybenko, George (1989). Approximation by superpositions of a sigmoidal function. *Mathematical Journal of Control, Signals, and Systemics*, 2, 303–314.

Deerwester, Scott C., Dumais, Susan T., Landauer, Thomas K., Furnas, George W. and Harshman, Richard A. (1990). Indexing by latent semantic analysis. *Journal of the American Society for Information Science*, 41, 391–407.

Eibe, Frank and Bouckaert, Remco (2006). Naïve Bayes for text classification with unbalanced classes. In *Lecture Notes in Computer Science*, Springer Berlin, pp. 503-510.

Eyheramendy, Susana, Lewis, David and Madigan, David (2003). On the naïve Bayes model for text categorization. In C. M. Bishop and B. J. Frey (Eds.), *Proceedings of the Ninth International Workshop on Artificial Intelligence and Statistics*.

Gao, Jing, Fan, Wei, Han, Jiawei and Yu, Philip S. (2007). A general framework for mining concept-drifting data streams with skewed distributions. *Proceedings of the Seventh SIAM International Conference on Data Mining (SDM-07)*, pp. 3–14.

Georgiopoulos, Michael, Huang, Juxin and Heileman, Gregory (1994). Properties of learning in ARTMAP. *Neural Networks*, 7(3), 495–506.

Grossberg, Stephen (1980). How does a brain build a cognitive code? *Psychological Review*, 87(1), 1–51.

Hastie, Trevor, Tibshirani, Robert and Friedman, Jerome (2001). *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer.

Hartman, Eric, Keeler, James D. and Kowalski, Jacek M. (1990). Layered neural networks with Gaussian hidden units as universal approximations. *Neural Computation,* 2(2), 210–215.

Hornik, Kurt, Stinchcombe, Maxwell and White, Halbert (1989). Multilayer feedforward networks are universal approximators, *Neural* Networks, 2(5), 359–366.

Joachims, Thorsten (1998). Text categorization with support vector machines: learning with many relevant features. In *Proceedings of the Tenth European Conference on Machine Learning (ECML-98)*, pp. 137–142.

Koprinska, Irena, Poon, Josiah, Clark, James and Chan, Jason (2007). Learning to classify e-mail. *Information Sciences,* 177(10), 2167–2187.

Manning, Christopher D. and Schütz, Hinrich (1999). *Foundations of Statistical Natural Language Processing*. MIT Press.

McCallum, Andrew and Nigam, Kamal (1998). A comparison of event models for naïve Bayes text classification. In *Proceedings of the AAAI-98 Workshop on Learning for Text Categorization*.

McRae, Ben (2005). Refresher guidance re licensing support network relevance and privilege designations, *Technical report,* U.S. Department of Energy, Washington, DC.

Minsky, Marvin and Papert, Seymour (1969). *Perceptrons: An Introduction to Computational Geometry*. The MIT Press.

Mitchell, Tom (1997). *Machine Learning*. McGraw-Hill.

Otis, Lee (2003). Screening and processing of licensing support network documentary material, *Technical report,* U.S. Department of Energy, Washington, DC.

Park, J. and Sandberg, I. W. (1991). Universal approximation using radial-basis-function networks. *Neural Computation*, 3(2), 246–257.

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

Quinlan, John Ross (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann.

Rennie, Jason (2001). Improving multi-class text classification with naïve Bayes. *MS Thesis*, Massachusetts Institute of Technology, Cambridge, MA.

Rosenblatt, Frank (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65:386–407.

Schneider, Karl-Michael (2005). Techniques for improving the performance of naïve Bayes for text classification. In *Computational Linguistics and Intelligent Text Processing*, pp. 682–693.

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

Shawe-Taylor, John and Nello Cristianini, Nello (2004). *Kernel Methods for Pattern Analysis*, Cambridge University Press.

Shen, Yirong and Jiang, Jing (2003). *Improving the Performance of Naïve Bayes for Text Classification*, Retrieved October 30, 2007, Class notes at Stanford University: http://nlp.stanford.edu/courses/cs224n/2003/fp/yirong99/report.pdf .

Verzi, Stephen, Heileman, Greg, Georgiopoulos, Michael and Anagnostopoulos, Georgios (2003). Universal function approximation with fuzzy ART and fuzzy ARTMAP. In *Proceedings of the IJCNN*, Vol. 3, pp. 1987–1992.

Verzi, Stephen (2003). Extensions to fuzzy ARTMAP based on structural risk minimization, *Ph.D. dissertation*, University of New Mexico, Albuquerque, NM.

WEKA (2007). *Weka Machine Learning Project*. Retrieved October 31, 2007, from Waikato Environment for Knowledge Analysis: http://www.cs.waikato.ac.nz/~ml/index.html.

Witten, Ian H. and Eibe, Frank (2005). *Data Mining: Practical Machine Learning Tools and Techniques*, 2nd Edition, Morgan Kaufmann.

This page intentionally left blank.

APPENDIX A. LSNAA SOFTWARE REQUIREMENTS

The list of requirements is taken from (Basilico and Verzi, 2006); the specific requirement numbers used in that document have been preserved here.

A.1 Input and Output Requirements

The LSN Assistant will be data-driven as directed by the user. Textual and visual outputs will be used as appropriate to display requested results.

A.1.1 Input

R3.2.1 – The archive tool shall allow the user to select the input documents.

R3.2.2 – Both the archive and the real-time tools shall allow the user to accept, reject, or modify a classification.

R3.2.3 – The system shall be able to accept files of the following formats: Plain Text (.txt), Microsoft Word Document (.doc), Portable Document Format (.pdf) (OCR only), and Microsoft PowerPoint Presentation (.ppt).

R3.2.4 – The archive tool shall be able to access documents and emails stored in a database.

R3.2.5 – The archive tool shall be able to access existing classifications for a document.

R3.2.6 – The system shall be able to accept email files in Lotus Notes and Microsoft Outlook formats, including the body text of the email and attachments that contain text in supported file types.

A.1.2 Output

R3.2.7 – Both the archive and real-time tools shall archive classifications so that they can be reviewed at a later date.

R3.2.8 – Both the archive and real-time tools shall provide a method for tracing the history of each document's classifications.

R3.2.9 – The archive tool shall output changed classifications so that they can be synchronized with the existing LSN support system.

A.2 Data Requirements

The LSN Assistant will store and process data to support the user's classification tasks. (Note: these data requirements do not include the external interfacing databases; see Appendix A.5.)

R3.3.1 – The archive tool shall allow the user to store and retrieve generated text reports.

A.3 Functional Requirements

R3.4.1 – The system shall classify a document based on available text.

R3.4.2 – The system's classification shall include the categorization of a document as relevant or non-relevant.

R3.4.3 – The system's classification shall include the categorization of a document as privileged or non-privileged, based on the training document sets provided.

R3.4.4 – For documents that are determined to be privileged, the system shall further determine the applicable privilege type, based on the training document sets provided.

R3.4.5 – The archive tool shall be able to display previous user-provided classifications of a document, including the original classification.

R3.4.6 – The archive tool shall be able to display the degree of discrepancy between the last archived user-provided classification and the recommended classification provided by the system, when there is a discrepancy between the two.

R3.4.7 – The archive tool shall allow for the user to provide a manual classification.

R3.4.8 – The archive tool shall generate text reports to detail, at a minimum, the time of review, the user conducting the review, the software and classifier versions, and the documents reviewed with relevant document header information, any previous classification, the recommended and chosen classification, and an explanation of how the classification was determined.

R3.4.9 – The archive tool shall allow the user to select the documents to be reviewed by specifying a date range.

R3.4.10 – The archive tool shall allow the user to select the documents to be reviewed by specifying the author/sender.

R3.4.11 – The archive tool shall allow the user to select the documents to be reviewed by specifying the author/sender's organization, if the organization information is available.

R3.4.12 – Both the archive and real-time tools shall indicate to the user if an email has an attachment from which the system cannot extract text.

R3.4.13 – The real-time tool shall review a document and recommend a classification. Users will then provide their own classification, which may or may not agree with the recommendation.

R3.4.14 – The system shall be capable of updating the classifier being used, subject to the customer's change control process.

R3.4.15 – The archive tool shall be able to display summary statistics about the document set to the user such as the total number of documents, the total number documents flagged has being a potential discrepancy, the number of documents in each category (relevant/non-relevant and privilege type), and the number of documents flagged as being a potential discrepancy in each category.

A.4 Performance Requirements

R3.5.1 – The system shall be able to generate the correct classification for a document at least 90% of the time for a designated set of example document classifications that are withheld from the building of the classifier.

R3.5.2 – The system shall be able to classify an average of at least 10 emails per second on the designated example data.

A.5 Systems and Communication Requirements

The LSN Assistant will interface with a variety of databases. Sandia organization 06783 will provide this access.

R3.6.1 – The archive tool shall be able to access the required email and document databases of archived documents to achieve the functional requirements through ODBC connections.

R3.6.2 – The real-time tool shall be able to interface with the existing network to perform classification in real-time.

R3.6.3 – The real-time tool shall be accessible as an integral part of the workspace for real-time classification.

A.6 System Security Requirements

No system security requirements (e.g., data classification, protection levels, access control, etc.) exist for the LSN Assistant. The LSN Assistant will rely on the security protection afforded by the host PC system.

A.7 Back up and Recovery Requirements

No continuity of operations is required in the event of a system failure. The LSN Assistant will rely on the back up operation of the host PC system for file back up.

A.8 Operating Environment Requirements

R3.9.1 – Both the archive and real-time tools shall operate on a PC with a minimum of 1GHz processor, 512 MB RAM, 1 GB free disk space, and 1024x768 screen resolution running a supported operating system with the ability to access the external databases referenced in 3.6 Systems and Communication Requirements

R3.9.2 – Both the archive and real-time tools shall operate under Windows XP.

R3.9.3 – Both the archive and real-time tools shall operate under Windows 2000.

A.9 Scalability and Reusability Requirements

The software shall be designed in a modular fashion to encourage reusability and extensibility wherever possible. This will be assessed during the design review.

R3.10.1 – The system design shall support a mechanism for adding support for accessing documents in new databases.

R3.10.2 – The system design shall support a mechanism for adding support for new document types.

A.10 Usability Requirements

The primary users of the LSN Assistant are the analysts in Sandia's Licensing Assessment and Technical Evaluation organization. These users are familiar with PC-based applications such as Microsoft Office and database search engines. Prototypes of the graphical user interface will be iterated upon early in the requirements and design phase based on user feedback. These prototype interfaces will evolve to the end product's graphical user interface design.

R3.11.1 – Microsoft Windows interface guidelines shall be used as a basis for the graphical interface design to ensure visual and functional consistency with other Windows-based applications.

R3.11.2 – The archive tool shall provide a user-validated graphical user interface for managing the classification of archived documents.

R3.11.3 – The real-time tool shall provide a user-validated graphical user interface that allows users to classify email in real-time.

R3.11.4 – A user's manual shall be developed for the real-time classification tool.

APPENDIX B. PREPARATION AND USE OF SCF CATEGORIZERS WITH THE LSNAA

The preparation and use of a categorizer in the context of the LSNAA is presented in this appendix. The steps performed in preparing the data for building a categorizer and for using the categorizer with a database designed to the specifications described in Section 2.4 are presented in Table 3. Detailed descriptions of these steps are presented in the individual sections of this appendix. The scripts listed in Table 3 are Windows PowerShell 1.0 scripts. These scripts reference SCF classes and methods as well as those available in the standard PowerShell environment.

Table 3. Steps for Preparation and Use of SCF Categorizers with the LSNAA.

Step	Description	Script(s)
1	Preparing data for use in the SCF	PrepareDataForCategorizer.ps1 SplitDocumentVectorDataSet.ps1
3	Determining individual categorizer parameters	ManualCategorizerOptimization.ps1
4	Building the SCF ensemble categorizer	BuildCategorizer_X.X.X.X.ps1
5	Validating the ensemble categorizer performance	CrossValidation.ps1
6	Updating the database categorizations	UpdateCategorizationsFromReviews.ps1

B.1 Preparing Data for Use in the SCF

Extracting the data from a database

The data to be used for training and testing the categorizer is extracted from the database and either used directly or stored in one or more files for subsequent processing. The determination should be made based on whether the extracted data should be stored for further processing or if the categorization building process will need to be performed more than once. The script denoted in Table 3 for this step contains code required for either type of processing.

In some cases, the data will need to be split into several files to facilitate the processing. The available RAM on the machine that will be preparing the data for use in the SCF will determine the maximum size of the data. There are no fixed rules for determine the number of files (i.e., partitions) the data will need to be split into.

As an example of the split requirements, we processed data from the BSC and Lead Lab (LL) e-mail database at two points. The machine used had 1Gb of RAM (approximately 460 Mb available for applications). At the first point, we processed 8 Mb of data (1766 e-mail messages) and did not need to split the data for processing. However, in the second step, we had 575 Mb of data (34,151 e-mail messages), and we had to split this data into 172 partitions of 200 e-mail messages each. Note that with a split of this data with 100 e-mail messages per split, we were not able to process the data.

Splitting each message into its own partitioning is not recommended, as this will dramatically increase the amount of time in processing the data. Users should always try to find the fewest number of partitions that will work for their data on their machines in order to maximize performance.

Building a model of the data for use in the SCF

The STANLEY text analysis library (Bauer, *et al*., 2005) is used for creating the data model. The data model is a vector space model of the data using *term frequency* for the local weighting and a *log-entropy* weighting for the global weighting. For all work documented in this report, the default settings of STANLEY were used with the exception that the data chunk size was set to 47 (see the STANLEY API documentation for more details).

Transforming the data for use in the SCF

Once the STANLEY data model has been created, it is applied to all of the data to be used in the SCF. We note that this step and the previous one are independent to allow for data models to be created from one set of data and applied to another. In our work , we created data models from validated data only, i.e., data that was manually validated for correctness by one or more human reviewers.

Splitting the validated data into sets for training and testing categorizers

Categorizers are built from one set of data and used to predict data class membership for other sets of data. Therefore, to validate the performance of categorizers in such a use case, testing must be performed on data that is not used to train those categorizers. Otherwise, the categorizer will be biased to perform well on the testing data, and the validation results will not be useful in evaluating categorizer performance. To accommodate this, the validated data is split into testing and training data sets. Most often this split partitions the data into equal-size partitions, but other splits are accommodated in the scripts.

B.2 Determining individual categorizer parameters

A set of parameters for the individual categorizers (i.e., the members of the ensemble categorizer) must be determined. Rigorous global optimization of the parameter space is costly and can lead to overfitting the training data, thus increasing the potential for reduced performance of applying the data models to testing data characteristically different that the training data. Thus, optimization over a discrete set of set of potentially useful parameters is recommended for determining individual categorizer parameters to be used. For this process, several values for each parameter of a given categorizer method are chosen and all combinations of these values across the full set of parameters are used to build an instance of a categorizer. The script denoted in Table 3 for this step provides the general framework for performing this process.

For this work, categorizers built using each of the methods and demonstrating the best performance on a set of testing data were used as members of the ensemble categorizer. In cases where categorizers with the same overall performance but characteristically difference types of behavior (e.g., one that finds a high percentage of the true positives versus one that finds a low percentage of false negatives), we opted to include instances of both categorizers in the ensemble.

B.3 Building the SCF categorizer

As a benchmark for evaluating performance of the ensemble categorizer and to provide data for optimizing the ensemble weights, an ensemble categorizer is built from the individual categorizers identified in the previous section. The ensemble weights (i.e., the relative contribution of each categorizer to the overall behavior of the ensemble categorizer) are initially set to 1.0, denoting equal contribution from each categorizer. However, these weights are adjustable to reflect any linear combination of contributions.

Within the SCF, a test of an ensemble categorizer consists of a test of each of the individual categorizers where the output is combined using the ensemble weights. Thus for a given set of test data, the individual categorizers need only be applied once, and the results saved. From that point, determination of the ensemble weights through more rigorous optimization can be made at a relatively low cost compared to building the model. A general approach to such an optimization of the weights can be challenging due to the competing objectives of increasing the number of true positive categorizations while reducing the number of false negative categorizations of the testing data. Often there is not a single set of weights that can be determined to be the best, and users must manually analyze the results of the optimization process to determine the set of ensemble weights to use.

B.4 Validating the categorizer performance

Once the parameters of the individual categorizers, the set of categorizers to be included in the ensemble categorizers, and the ensemble weights have been determined, it is recommended to assess the performance and robustness of the resulting ensemble categorizer. This process of validating the categorizer consists of breaking a set of validated data (i.e., data with categorizations manually checked by one or more human reviewers) into partitions, building several ensemble categorizers using subsets of the data, and testing each of the resulting ensemble categorizers with the remainder of the data not used in training the categorizer. The statistics of the results of these tests can be used to access the performance and robustness of the ensemble categorizers.

In the SCF, there are two types of categorizer validation that are supported: *K*-fold cross validation and random split validation. *K*-fold cross validation produces *K* folds (partitions) of the validated data with approximately equal size folds. Each fold is used as a testing set for a categorizer built using all other data as the training set. The testing output is averaged across the folds. The standard deviation of the test results from the different folds reflects how robust the

ensemble performance over a range of different data sets. Random split validation produces *N* splits of the data, each with approximately equal size folds. The percentage of data points in each set is chosen randomly from a Gaussian distribution. As *N* increases, this leads to an average of equal sized splits with very small standard deviation. For each split, one set is used to train a categorizer, and the other is used for testing that categorizer. The testing output is averaged across the splits.

B.5 Updating the database categorizations

Once the ensemble categorizer has been created and its performance validated, the final step is to apply the categorizer to a set of e-mail messages residing in an in an email categorization database (see Section 2.3).

Connecting to the database, loading of the ensemble categorizers, retrieving of the e-mail message data, categorizing the data, and posting of the new categorizations to the database are facilitated using the script denoted in Table 3 for this step. The script accommodates connecting to different servers (one at a time) and generates categorizations for all records in the database (i.e., there is currently no option to categorize subsets of records using the SCF). Each set of categorizations is created and posted to the database is archived and tagged with both a user-defined categorizer name and internal version number, so that each categorization can be retrieved and used independently. Note that the LSN Archive Assistant uses the most recent categorizations available in each database by default.

APPENDIX C. RESULTS OF THE WEKA CATEGORIZERS

C.1 The Parameters used to Build the WEKA Categorizers

Table 4 presents the parameters used to build the WEKA categorizers used in determining the LSNAA algorithms. Each categorizer name is a short description of the algorithm, and algorithms tested more than once with different parameters are listed with different numbers. The parameters shown here are those specified in WEKA 3.5.5.

Table 4. Parameters used to Build the WEKA Categorizers.

Categorizer Algorithm Name	Type	Parameters
Alternating Decision Tree 1	Decision Tree	ADTree -B 10 -E 0
Alternating Decision Tree 2	Decision Tree	ADTree -B 10 -E -1
Alternating Decision Tree 3	Decision Tree	ADTree -B 10 -E -2
Alternating Decision Tree 4	Decision Tree	ADTree -B 10 -E -3
Alternating Decision Tree 5	Decision Tree	ADTree -B 20 -E 0
Alternating Decision Tree 6	Decision Tree	ADTree -B 30 -E 0
Alternating Decision Tree 7	Decision Tree	ADTree -B 40 -E 0
Alternating Decision Tree 8	Decision Tree	ADTree -B 50 -E 0
Bagging 1	Meta-learner	Bagging -P 10 -S 1 -I 10 -W weka.classifiers.misc.VFI -- -B 0.6
Bagging 2	Meta-learner	Bagging -P 100 -S 1 -I 10 -W weka.classifiers.misc.VFI -- -B 0.6
Bagging 3	Meta-learner	Bagging -P 80 -S 1 -I 30 -W weka.classifiers.misc.VFI -- -B 0.01
Bayes Network Learning Algorithm	Bayesian	BayesNet -D -Q bayes.net.search.local.K2 -- -P 1 -S BAYES -E bayes.net.estimate.SimpleEstimator -- -A 0.5
Best-First Decision Tree 1	Decision Tree	BFTree -S 1 -M 2 -N 5 -G -R -C 1.0 -P UNPRUNED
Best-First Decision Tree 2	Decision Tree	BFTree -S 1 -M 2 -N 5 -R -C 1.0 -P UNPRUNED
Dagging 1	Meta-learner	Dagging -F 10 -S 1 -W weka.classifiers.functions.SMO -- -C 1.0 -L 0.0010 -P
Dagging 2	Meta-learner	Dagging -F 10 -S 1 -W weka.classifiers.misc.VFI -- -B 0.6
Decision Stump	Decision Tree	DecisionStump
Decision Table	Decision Rule	DecisionTable -X 1 -S "weka.attributeSelection.BestFirst -D 1 -N 5"
Grading	Meta-learner	Grading -X 10 -M "weka.classifiers.rules.ZeroR " -S 1 -B "weka.classifiers.trees.RandomTree -K 20 -M 1.0 -S 1" (times 4)
HyperPipes	Miscellaneous	HyperPipes
Nearest Neighbor 1	Lazy	IB1
Nearest Neighbor 2	Lazy	IBk -K 10 -W 0 -A "weka.core.LinearNN -A weka.core.EuclideanDistance"

Categorizer Algorithm Name	Type	Parameters
Nearest Neighbor 3	Lazy	IBk -K 5 -W 0 -A "weka.core.LinearNN -A weka.core.EuclideanDistance"
C4.5 Decision Tree 1	Decision Tree	J48 -C 0.25 -B -M 2
C4.5 Decision Tree 2	Decision Tree	J48 -C 0.25 -M 2
C4.5 Decision Tree 3	Decision Tree	J48 -C 0.25 -M 2 -A
C4.5 Decision Tree 4	Decision Tree	J48 -U -M 2
Ripper 1	Decision Rule	JRip -F 3 -N 2.0 -O 2 -S 1
Ripper 2	Decision Rule	JRip -F 3 -N 2.0 -O 2 -S 1 -E
Ripper 3	Decision Rule	JRip -F 3 -N 2.0 -O 2 -S 1 -P
Ripper 4	Decision Rule	JRip -F 3 -N 2.0 -O 5 -S 1
Logistic Model Decision Tree	Decision Tree	LMT -I -1 -M 15 -W 0.0
Multinomial Logistic Regression 1	Function	Logistic -R 1.0E-8 -M -1
Multinomial Logistic Regression 2	Function	Logistic -R 1.0E-8 -M -1
Boosting + Wagging	Meta-learner	MultiBoostAB -C 5 -S 1 -I 10 -W weka.classifiers.misc.VFI -- -B 0.1 -Q -P 100
Naïve Bayes	Bayesian	NaiveBayes
Naïve Bayes Decision Tree	Decision Tree	NBTree
Nearest Neighbor with Generalization	Decision Rule	NNge -G 5 -I 5
1R 1	Decision Rule	OneR -B 1
1R 2	Decision Rule	OneR -B 2
1R 3	Decision Rule	OneR -B 3
1R 4	Decision Rule	OneR -B 5
Partial C4.5 Decision Tree Rules 1	Decision Rule	PART -B -M 2 -C 0.25 -Q 1
Partial C4.5 Decision Tree Rules 2	Decision Rule	PART -M 2 -C 0.5 -Q 1
Partial C4.5 Decision Tree Rules 3	Decision Rule	PART -R -M 2 -N 3 -Q 1
Partial C4.5 Decision Tree Rules 4	Decision Rule	PART -U -M 2 -C 0.25 -Q 1
Random Committee	Meta-learner	RandomCommittee -S 1 -I 10 -W weka.classifiers.trees.RandomTree -- -K 1 -M 1.0 -S 1
Random Forest 1	Decision Tree	RandomForest -I 10 -K 0 –S 1
Random Forest 2	Decision Tree	RandomForest -I 10 -K 10 -S 1
Random Forest 3	Decision Tree	RandomForest -I 10 -K 20 -S 1
Random Subspace	Meta-learner	RandomSubSpace -P 0.5 -S 1 -I 10 -W weka.classifiers.bayes.NaiveBayes --
Random Decision Tree 1	Decision Tree	RandomTree -K 1 -M 1.0 -S 1
Random Decision Tree 2	Decision Tree	RandomTree -K 10 -M 1.0 -S 1
Random Decision Tree 3	Decision Tree	RandomTree -K 20 -M 1.0 -S 1
Random Decision Tree 4	Decision Tree	RandomTree -K 25 -M 1.0 -S 1
Random Decision Tree 5	Decision Tree	RandomTree -K 30 -M 1.0 -S 1
Random Decision Tree 6	Decision Tree	RandomTree -K 50 -M 1.0 -S 1
Radial Basis Function Network	Function	RBFNetwork -B 2 -S 1 -R 1.0E-8 -M -1 -W 0.1
Reduced Error Pruning Tree 1	Decision Tree	REPTree -M 2 -V 0.0010 -N 3 -S 1 -L -1
Reduced Error Pruning Tree 2	Decision Tree	REPTree -M 2 -V 0.0010 -N 3 -S 1 -L -1 -P
Reduced Error Pruning Tree 3	Decision Tree	REPTree -M 2 -V 0.0010 -N 3 -S 1 -L 10 -P
Ripple Down Rule Learner 1	Decision Rule	Ridor -F 3 -S 1 -N 2.0
Ripple Down Rule Learner 2	Decision Rule	Ridor -F 3 -S 1 -N 2.0 -A
Regression Decision Tree with Pruning	Decision Tree	SimpleCart -S 1 -M 2.0 -N 5 -U -C 1.0
Support Vector Classifier	Function	SMO -C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1 -K

Categorizer Algorithm Name	Type	Parameters
Stacking	Meta-learner	Stacking -X 10 -M "weka.classifiers.rules.ZeroR " -S 1 -B "weka.classifiers.bayes.NaiveBayes " -B "weka.classifiers.misc.VFI -B 0.01" -B "weka.classifiers.misc.HyperPipes "
Voted Feature Intervals 1	Miscellaneous	VFI -B 0.01
Voted Feature Intervals 2	Miscellaneous	VFI -B 0.1
Voted Feature Intervals 3	Miscellaneous	VFI -B 0.2
Voted Feature Intervals 4	Miscellaneous	VFI -B 0.6
Voted Feature Intervals 5	Miscellaneous	VFI -B 0.9
Voted Feature Intervals 6	Miscellaneous	VFI -C -B 0.6
Voted Ensemble	Meta-learner	Vote -B "weka.classifiers.misc.VFI -B 0.1" -B "weka.classifiers.misc.HyperPipes " -B "DecisionStump " -B "RandomTree -K 1 -M 1.0 -S 1" -B "NNge -G 5 -I 5" -B "NaiveBayes " -B "ZeroR " -R AVG
Voted Perceptron	Meta-learner	VotedPerceptron -I 1 -E 1.0 -S 1 -M 10000
0R	Decision Rule	ZeroR

C.2 The Results of the WEKA Categorizers

Table 5 presents the results of the WEKA categorizers built using the parameters in Table 4. In binary categorization, the classes are labeled *positive* and *negative,* and thus we report here the numbers of true negatives (TN), false negatives (FN), true positives (TP), and false positives (FP) for each categorizer. We also report three standard measures for evaluating the performance of each category: *accuracy* (percent correct over both classes), *precision* (the percentage of positives marked correctly), and *recall* (the percentage of positives marked positive by the categorizer). Note there are 4548 positives and 165 negatives in this training set.

Table 5. Results of the WEKA Categorizers.

Categorizer Algorithm Name	TN	FN	TP	FP	Accuracy	Precision	Recall
Alternating Decision Tree 1	4511	151	14	37	96.01	0.27	0.08
Alternating Decision Tree 2	4524	157	8	24	96.16	0.25	0.05
Alternating Decision Tree 3	4524	157	8	24	96.16	0.25	0.05
Alternating Decision Tree 4	4548	165	0	0	96.50	0.00	0.00
Alternating Decision Tree 5	4508	136	29	40	96.27	0.42	0.18
Alternating Decision Tree 6	4512	132	33	36	96.44	0.48	0.20
Alternating Decision Tree 7	4509	131	34	39	96.39	0.47	0.21
Alternating Decision Tree 8	4508	130	35	40	96.39	0.47	0.21
Bagging 1	4333	144	21	215	92.38	0.09	0.13
Bagging 2	3018	46	119	1530	66.56	0.07	0.72
Bagging 3	2475	33	132	2073	55.32	0.06	0.80
Bayes Network Learning Algorithm	4403	105	60	145	94.70	0.29	0.36
Best-First Decision Tree 1	4447	124	41	101	95.23	0.29	0.25
Best-First Decision Tree 2	4526	139	26	22	96.58	0.54	0.16
Dagging 1	4548	152	13	0	96.77	1.00	0.08
Dagging 2	4196	133	32	352	89.71	0.08	0.19
Decision Stump	4548	165	0	0	96.50	0.00	0.00
Decision Table	4520	148	17	28	96.27	0.38	0.10

Categorizer Algorithm Name	TN	FN	TP	FP	Accuracy	Precision	Recall
Grading	4521	130	35	27	96.67	0.56	0.21
HyperPipes	3567	86	79	981	77.36	0.07	0.48
Nearest Neighbor 1	4475	100	65	73	96.33	0.47	0.39
Nearest Neighbor 2	4531	134	31	17	96.80	0.65	0.19
Nearest Neighbor 3	4522	130	35	26	96.69	0.57	0.21
C4.5 Decision Tree 1	4451	128	37	97	95.23	0.28	0.22
C4.5 Decision Tree 2	4451	128	37	97	95.23	0.28	0.22
C4.5 Decision Tree 3	4451	128	37	97	95.23	0.28	0.22
C4.5 Decision Tree 4	4447	128	37	101	95.14	0.27	0.22
Ripper 1	4502	144	21	46	95.97	0.31	0.13
Ripper 2	4502	145	20	46	95.95	0.30	0.12
Ripper 3	4520	131	34	28	96.41	0.55	0.21
Ripper 4	4504	144	21	44	96.01	0.32	0.13
Logistic Model Decision Tree	4535	148	17	13	96.58	0.57	0.10
Multinomial Logistic Regression 1	4451	118	47	97	95.44	0.33	0.28
Multinomial Logistic Regression 2	4451	118	47	97	95.44	0.33	0.28
Boosting + Wagging	2733	39	126	1815	60.66	0.06	0.76
Naïve Bayes	4341	93	72	207	93.63	0.26	0.44
Naïve Bayes Decision Tree	4521	137	28	27	96.52	0.51	0.17
Nearest Neighbor with Generalization	4545	148	17	3	96.80	0.85	0.10
1R 1	4397	160	5	151	93.40	0.03	0.03
1R 2	4524	160	5	24	96.10	0.17	0.03
1R 3	4521	156	9	27	96.12	0.25	0.05
1R 4	4533	161	4	15	96.27	0.21	0.02
Partial C4.5 Decision Tree Rules 1	4454	127	38	94	95.31	0.29	0.23
Partial C4.5 Decision Tree Rules 2	4454	127	38	94	95.31	0.29	0.23
Partial C4.5 Decision Tree Rules 3	4508	145	20	40	96.07	0.33	0.12
Partial C4.5 Decision Tree Rules 4	4454	124	41	94	95.37	0.30	0.25
Random Committee	4541	140	25	7	96.88	0.78	0.15
Random Forest 1	4536	131	34	12	96.97	0.74	0.21
Random Forest 2	4539	134	31	9	96.97	0.78	0.19
Random Forest 3	4538	137	28	10	96.88	0.74	0.17
Random Subspace	4439	103	62	109	95.50	0.36	0.38
Random Decision Tree 1	4433	131	34	115	94.78	0.23	0.21
Random Decision Tree 2	4421	123	42	127	94.70	0.25	0.25
Random Decision Tree 3	4463	125	40	85	95.54	0.32	0.24
Random Decision Tree 4	4487	121	44	61	96.14	0.42	0.27
Random Decision Tree 5	4458	124	41	90	95.46	0.31	0.25
Random Decision Tree 6	4452	126	39	96	95.29	0.29	0.24
Radial Basis Function Network	4548	165	0	0	96.50	0.00	0.00
Reduced Error Pruning Tree 1	4548	165	0	0	96.50	0.00	0.00
Reduced Error Pruning Tree 2	4448	124	41	100	95.25	0.29	0.25
Reduced Error Pruning Tree 3	4473	132	33	75	95.61	0.31	0.20
Ripple Down Rule Learner 1	4538	154	11	10	96.52	0.52	0.07
Ripple Down Rule Learner 2	4427	114	51	121	95.01	0.30	0.31
Regression Decision Tree with Pruning	4526	139	26	22	96.58	0.54	0.16
Support Vector Classifier	4538	138	27	10	96.86	0.73	0.16
Stacking	4548	165	0	0	96.50	0.00	0.00
Voted Feature Intervals 1	2244	24	141	2304	50.60	0.06	0.85
Voted Feature Intervals 2	2325	26	139	2223	52.28	0.06	0.84
Voted Feature Intervals 3	2415	30	135	2133	54.11	0.06	0.82
Voted Feature Intervals 4	2731	39	126	1817	60.62	0.06	0.76

Categorizer Algorithm Name	TN	FN	TP	FP	Accuracy	Precision	Recall
Voted Feature Intervals 5	2953	48	117	1595	65.14	0.07	0.71
Voted Feature Intervals 6	2227	24	141	2321	50.24	0.06	0.85
Voted Ensemble	4547	150	15	1	96.80	0.94	0.09
Voted Perceptron	4538	144	21	10	96.73	0.68	0.13
0R	4548	165	0	0	96.50	0.00	0.00

This page intentionally left blank.

APPENDIX D. RESULTS OF THE CATEGORIZERS IN THE LSNAA

D.1 Individual Categorizers: First Review

Table 6 presents the results of the *Perceptron Categorizer*, where P_1 denotes the maximum number of iterations, P_2 denotes the margin of positive instances, and P_3 denotes the margin of negative instances.

Table 6. Perceptron Categorizer Results: First Review.

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
100	1	1	15	831	6	31	0.958	0.714	0.326
100	1	10	12	835	2	34	0.959	0.857	0.261
100	1	50	9	835	2	37	0.956	0.818	0.196
100	1	100	5	835	2	41	0.951	0.714	0.109
100	10	1	21	808	29	25	0.939	0.420	0.457
100	10	10	15	831	6	31	0.958	0.714	0.326
100	10	50	12	835	2	34	0.959	0.857	0.261
100	10	100	6	835	2	40	0.952	0.750	0.130
100	50	1	32	775	62	14	0.914	0.340	0.696
100	50	10	24	812	25	22	0.947	0.490	0.522
100	50	50	14	832	5	32	0.958	0.737	0.304
100	50	100	6	835	2	40	0.952	0.750	0.130
100	100	1	38	710	127	8	0.847	0.230	0.826
100	100	10	30	783	54	16	0.921	0.357	0.652
100	100	50	14	829	8	32	0.955	0.636	0.304
100	100	100	6	835	2	40	0.952	0.750	0.130
500	1	1	15	831	6	31	0.958	0.714	0.326
500	1	10	12	834	3	34	0.958	0.800	0.261
500	1	50	10	835	2	36	0.957	0.833	0.217
500	1	100	8	835	2	38	0.955	0.800	0.174
500	10	1	23	808	29	23	0.941	0.442	0.500
500	10	10	15	830	7	31	0.957	0.682	0.326
500	10	50	12	835	2	34	0.959	0.857	0.261
500	10	100	12	835	2	34	0.959	0.857	0.261
500	50	1	32	776	61	14	0.915	0.344	0.696
500	50	10	23	808	29	23	0.941	0.442	0.500
500	50	50	15	831	6	31	0.958	0.714	0.326
500	50	100	13	831	6	33	0.956	0.684	0.283
500	100	1	34	762	75	12	0.901	0.312	0.739
500	100	10	30	790	47	16	0.929	0.390	0.652
500	100	50	18	827	10	28	0.957	0.643	0.391
500	100	100	15	831	6	31	0.958	0.714	0.326
1000	1	1	15	831	6	31	0.958	0.714	0.326
1000	1	10	12	834	3	34	0.958	0.800	0.261
1000	1	50	8	835	2	38	0.955	0.800	0.174
1000	1	100	8	835	2	38	0.955	0.800	0.174
1000	10	1	23	808	29	23	0.941	0.442	0.500
1000	10	10	15	830	7	31	0.957	0.682	0.326
1000	10	50	12	834	3	34	0.958	0.800	0.261
1000	10	100	12	835	2	34	0.959	0.857	0.261

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
1000	50	1	32	774	63	14	0.913	0.337	0.696
1000	50	10	25	807	30	21	0.942	0.455	0.543
1000	50	50	15	830	7	31	0.957	0.682	0.326
1000	50	100	14	831	6	32	0.957	0.700	0.304
1000	100	1	32	768	69	14	0.906	0.317	0.696
1000	100	10	30	793	44	16	0.932	0.405	0.652
1000	100	50	17	827	10	29	0.956	0.630	0.370
1000	100	100	15	831	6	31	0.958	0.714	0.326
5000	1	1	15	831	6	31	0.958	0.714	0.326
5000	1	10	12	834	3	34	0.958	0.800	0.261
5000	1	50	8	835	2	38	0.955	0.800	0.174
5000	1	100	7	835	2	39	0.954	0.778	0.152
5000	10	1	23	808	29	23	0.941	0.442	0.500
5000	10	10	15	830	7	31	0.957	0.682	0.326
5000	10	50	12	834	3	34	0.958	0.800	0.261
5000	10	100	12	835	2	34	0.959	0.857	0.261
5000	50	1	32	774	63	14	0.913	0.337	0.696
5000	50	10	25	807	30	21	0.942	0.455	0.543
5000	50	50	15	829	8	31	0.956	0.652	0.326
5000	50	100	14	830	7	32	0.956	0.667	0.304
5000	100	1	32	768	69	14	0.906	0.317	0.696
5000	100	10	29	796	41	17	0.934	0.414	0.630
5000	100	50	17	826	11	29	0.955	0.607	0.370
5000	100	100	15	829	8	31	0.956	0.652	0.326

Table 7 presents the results of the *Balanced Perceptron Categorizer*, where P_1 denotes the maximum number of iterations, P_2 denotes the margin of positive instances, P_3 denotes the margin of negative instances, P_4 denotes the number of balanced learners, and P_5 denotes the majority class percentage used to train each learner.

Table 7. Balanced Perceptron Categorizer Results: First Review.

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
100	50	1	1	1.50	45	148	689	1	0.219	0.061	0.978
100	50	1	1	2.00	43	222	615	3	0.300	0.065	0.935
100	50	1	1	2.50	43	202	635	3	0.277	0.063	0.935
100	50	1	5	1.50	45	120	717	1	0.187	0.059	0.978
100	50	1	5	2.00	44	179	658	2	0.253	0.063	0.957
100	50	1	5	2.50	43	210	627	3	0.287	0.064	0.935
100	50	1	10	1.50	45	132	705	1	0.200	0.060	0.978
100	50	1	10	2.00	44	169	668	2	0.241	0.062	0.957
100	50	1	10	2.50	43	226	611	3	0.305	0.066	0.935
100	50	10	1	1.50	44	258	579	2	0.342	0.071	0.957
100	50	10	1	2.00	42	359	478	4	0.454	0.081	0.913
100	50	10	1	2.50	43	404	433	3	0.506	0.090	0.935
100	50	10	5	1.50	43	304	533	3	0.393	0.075	0.935
100	50	10	5	2.00	42	397	440	4	0.497	0.087	0.913
100	50	10	5	2.50	42	436	401	4	0.541	0.095	0.913
100	50	10	10	1.50	43	312	525	3	0.402	0.076	0.935
100	50	10	10	2.00	42	379	458	4	0.477	0.084	0.913

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
100	50	10	10	2.50	42	439	398	4	0.545	0.095	0.913
100	100	1	1	1.50	46	96	741	0	0.161	0.058	1.000
100	100	1	1	2.00	44	119	718	2	0.185	0.058	0.957
100	100	1	1	2.50	43	170	667	3	0.241	0.061	0.935
100	100	1	5	1.50	46	83	754	0	0.146	0.058	1.000
100	100	1	5	2.00	45	121	716	1	0.188	0.059	0.978
100	100	1	5	2.50	43	142	695	3	0.210	0.058	0.935
100	100	1	10	1.50	46	87	750	0	0.151	0.058	1.000
100	100	1	10	2.00	45	116	721	1	0.182	0.059	0.978
100	100	1	10	2.50	43	141	696	3	0.208	0.058	0.935
100	100	10	1	1.50	45	152	685	1	0.223	0.062	0.978
100	100	10	1	2.00	43	196	641	3	0.271	0.063	0.935
100	100	10	1	2.50	42	239	598	4	0.318	0.066	0.913
100	100	10	5	1.50	45	167	670	1	0.240	0.063	0.978
100	100	10	5	2.00	43	201	636	3	0.276	0.063	0.935
100	100	10	5	2.50	42	241	596	4	0.320	0.066	0.913
100	100	10	10	1.50	44	156	681	2	0.227	0.061	0.957
100	100	10	10	2.00	43	197	640	3	0.272	0.063	0.935
100	100	10	10	2.50	42	252	585	4	0.333	0.067	0.913
500	50	1	1	1.50	45	113	724	1	0.179	0.059	0.978
500	50	1	1	2.00	45	129	708	1	0.197	0.060	0.978
500	50	1	1	2.50	42	201	636	4	0.275	0.062	0.913
500	50	1	5	1.50	45	103	734	1	0.168	0.058	0.978
500	50	1	5	2.00	45	141	696	1	0.211	0.061	0.978
500	50	1	5	2.50	42	212	625	4	0.288	0.063	0.913
500	50	1	10	1.50	45	117	720	1	0.183	0.059	0.978
500	50	1	10	2.00	45	141	696	1	0.211	0.061	0.978
500	50	1	10	2.50	42	211	626	4	0.287	0.063	0.913
500	50	10	1	1.50	43	253	584	3	0.335	0.069	0.935
500	50	10	1	2.00	43	351	486	3	0.446	0.081	0.935
500	50	10	1	2.50	42	412	425	4	0.514	0.090	0.913
500	50	10	5	1.50	43	267	570	3	0.351	0.070	0.935
500	50	10	5	2.00	43	324	513	3	0.416	0.077	0.935
500	50	10	5	2.50	42	409	428	4	0.511	0.089	0.913
500	50	10	10	1.50	43	259	578	3	0.342	0.069	0.935
500	50	10	10	2.00	43	331	506	3	0.424	0.078	0.935
500	50	10	10	2.50	42	408	429	4	0.510	0.089	0.913
500	100	1	1	1.50	45	89	748	1	0.152	0.057	0.978
500	100	1	1	2.00	45	115	722	1	0.181	0.059	0.978
500	100	1	1	2.50	44	167	670	2	0.239	0.062	0.957
500	100	1	5	1.50	45	96	741	1	0.160	0.057	0.978
500	100	1	5	2.00	45	112	725	1	0.178	0.058	0.978
500	100	1	5	2.50	45	142	695	1	0.212	0.061	0.978
500	100	1	10	1.50	45	94	743	1	0.157	0.057	0.978
500	100	1	10	2.00	45	112	725	1	0.178	0.058	0.978
500	100	1	10	2.50	44	146	691	2	0.215	0.060	0.957
500	100	10	1	1.50	45	168	669	1	0.241	0.063	0.978
500	100	10	1	2.00	45	189	648	1	0.265	0.065	0.978
500	100	10	1	2.50	43	260	577	3	0.343	0.069	0.935
500	100	10	5	1.50	45	164	673	1	0.237	0.063	0.978
500	100	10	5	2.00	43	203	634	3	0.279	0.064	0.935
500	100	10	5	2.50	43	226	611	3	0.305	0.066	0.935

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
500	100	10	10	1.50	45	162	675	1	0.234	0.063	0.978
500	100	10	10	2.00	44	197	640	2	0.273	0.064	0.957
500	100	10	10	2.50	43	245	592	3	0.326	0.068	0.935
1000	50	1	1	1.50	45	116	721	1	0.182	0.059	0.978
1000	50	1	1	2.00	45	128	709	1	0.196	0.060	0.978
1000	50	1	1	2.50	43	192	645	3	0.266	0.063	0.935
1000	50	1	5	1.50	45	112	725	1	0.178	0.058	0.978
1000	50	1	5	2.00	45	140	697	1	0.210	0.061	0.978
1000	50	1	5	2.50	42	205	632	4	0.280	0.062	0.913
1000	50	1	10	1.50	45	111	726	1	0.177	0.058	0.978
1000	50	1	10	2.00	45	135	702	1	0.204	0.060	0.978
1000	50	1	10	2.50	43	195	642	3	0.270	0.063	0.935
1000	50	10	1	1.50	43	260	577	3	0.343	0.069	0.935
1000	50	10	1	2.00	43	318	519	3	0.409	0.077	0.935
1000	50	10	1	2.50	42	377	460	4	0.475	0.084	0.913
1000	50	10	5	1.50	43	262	575	3	0.345	0.070	0.935
1000	50	10	5	2.00	43	319	518	3	0.410	0.077	0.935
1000	50	10	5	2.50	42	384	453	4	0.482	0.085	0.913
1000	50	10	10	1.50	43	270	567	3	0.354	0.070	0.935
1000	50	10	10	2.00	43	333	504	3	0.426	0.079	0.935
1000	50	10	10	2.50	42	383	454	4	0.481	0.085	0.913
1000	100	1	1	1.50	45	105	732	1	0.170	0.058	0.978
1000	100	1	1	2.00	45	108	729	1	0.173	0.058	0.978
1000	100	1	1	2.50	43	163	674	3	0.233	0.060	0.935
1000	100	1	5	1.50	45	99	738	1	0.163	0.057	0.978
1000	100	1	5	2.00	45	113	724	1	0.179	0.059	0.978
1000	100	1	5	2.50	43	179	658	3	0.251	0.061	0.935
1000	100	1	10	1.50	45	98	739	1	0.162	0.057	0.978
1000	100	1	10	2.00	45	114	723	1	0.180	0.059	0.978
1000	100	1	10	2.50	43	172	665	3	0.243	0.061	0.935
1000	100	10	1	1.50	45	149	688	1	0.220	0.061	0.978
1000	100	10	1	2.00	43	192	645	3	0.266	0.063	0.935
1000	100	10	1	2.50	42	274	563	4	0.358	0.069	0.913
1000	100	10	5	1.50	45	159	678	1	0.231	0.062	0.978
1000	100	10	5	2.00	44	194	643	2	0.270	0.064	0.957
1000	100	10	5	2.50	42	258	579	4	0.340	0.068	0.913
1000	100	10	10	1.50	45	160	677	1	0.232	0.062	0.978
1000	100	10	10	2.00	44	204	633	2	0.281	0.065	0.957
1000	100	10	10	2.50	42	277	560	4	0.361	0.070	0.913
5000	50	1	1	1.50	45	108	729	1	0.173	0.058	0.978
5000	50	1	1	2.00	45	128	709	1	0.196	0.060	0.978
5000	50	1	1	2.50	42	208	629	4	0.283	0.063	0.913
5000	50	1	5	1.50	45	113	724	1	0.179	0.059	0.978
5000	50	1	5	2.00	45	144	693	1	0.214	0.061	0.978
5000	50	1	5	2.50	43	191	646	3	0.265	0.062	0.935
5000	50	1	10	1.50	45	120	717	1	0.187	0.059	0.978
5000	50	1	10	2.00	45	138	699	1	0.207	0.060	0.978
5000	50	1	10	2.50	43	195	642	3	0.270	0.063	0.935
5000	50	10	1	1.50	43	265	572	3	0.349	0.070	0.935
5000	50	10	1	2.00	43	348	489	3	0.443	0.081	0.935
5000	50	10	1	2.50	42	360	477	4	0.455	0.081	0.913
5000	50	10	5	1.50	43	262	575	3	0.345	0.070	0.935

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
5000	50	10	5	2.00	43	321	516	3	0.412	0.077	0.935
5000	50	10	5	2.50	42	406	431	4	0.507	0.089	0.913
5000	50	10	10	1.50	43	268	569	3	0.352	0.070	0.935
5000	50	10	10	2.00	43	329	508	3	0.421	0.078	0.935
5000	50	10	10	2.50	42	391	446	4	0.490	0.086	0.913
5000	100	1	1	1.50	46	97	740	0	0.162	0.059	1.000
5000	100	1	1	2.00	45	107	730	1	0.172	0.058	0.978
5000	100	1	1	2.50	43	195	642	3	0.270	0.063	0.935
5000	100	1	5	1.50	45	98	739	1	0.162	0.057	0.978
5000	100	1	5	2.00	45	115	722	1	0.181	0.059	0.978
5000	100	1	5	2.50	43	172	665	3	0.243	0.061	0.935
5000	100	1	10	1.50	45	97	740	1	0.161	0.057	0.978
5000	100	1	10	2.00	45	115	722	1	0.181	0.059	0.978
5000	100	1	10	2.50	43	177	660	3	0.249	0.061	0.935
5000	100	10	1	1.50	45	160	677	1	0.232	0.062	0.978
5000	100	10	1	2.00	44	196	641	2	0.272	0.064	0.957
5000	100	10	1	2.50	42	250	587	4	0.331	0.067	0.913
5000	100	10	5	1.50	45	162	675	1	0.234	0.063	0.978
5000	100	10	5	2.00	43	209	628	3	0.285	0.064	0.935
5000	100	10	5	2.50	42	248	589	4	0.328	0.067	0.913
5000	100	10	10	1.50	45	160	677	1	0.232	0.062	0.978
5000	100	10	10	2.00	44	202	635	2	0.279	0.065	0.957
5000	100	10	10	2.50	42	259	578	4	0.341	0.068	0.913

Table 8 presents the results of the *Balanced Random Forest Categorizer*, where P_1 denotes the number of random trees, P_2 denotes the number of random attributes, and P_3 denotes the majority class percentage used to train each tree.

Table 8. Balanced Random Forest Categorizer Results: First Review.

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
10	25	1.50	23	741	96	23	0.865	0.193	0.500
10	25	2.00	21	768	69	25	0.894	0.233	0.457
10	25	2.50	25	788	49	21	0.921	0.338	0.543
10	50	1.50	26	759	78	20	0.889	0.250	0.565
10	50	2.00	23	765	72	23	0.892	0.242	0.500
10	50	2.50	21	796	41	25	0.925	0.339	0.457
10	100	1.50	26	728	109	20	0.854	0.193	0.565
10	100	2.00	23	745	92	23	0.870	0.200	0.500
10	100	2.50	23	763	74	23	0.890	0.237	0.500
10	200	1.50	26	749	88	20	0.878	0.228	0.565
10	200	2.00	25	750	87	21	0.878	0.223	0.543
10	200	2.50	18	777	60	28	0.900	0.231	0.391
25	25	1.50	24	749	88	22	0.875	0.214	0.522
25	25	2.00	23	759	78	23	0.886	0.228	0.500
25	25	2.50	20	805	32	26	0.934	0.385	0.435
25	50	1.50	27	755	82	19	0.886	0.248	0.587
25	50	2.00	22	739	98	24	0.862	0.183	0.478
25	50	2.50	19	807	30	27	0.935	0.388	0.413
25	100	1.50	30	736	101	16	0.867	0.229	0.652

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
25	100	2.00	25	753	84	21	0.881	0.229	0.543
25	100	2.50	20	787	50	26	0.914	0.286	0.435
25	200	1.50	31	716	121	15	0.846	0.204	0.674
25	200	2.00	24	773	64	22	0.903	0.273	0.522
25	200	2.50	21	785	52	25	0.913	0.288	0.457
50	25	1.50	24	736	101	22	0.861	0.192	0.522
50	25	2.00	23	774	63	23	0.903	0.267	0.500
50	25	2.50	25	784	53	21	0.916	0.321	0.543
50	50	1.50	25	743	94	21	0.870	0.210	0.543
50	50	2.00	25	759	78	21	0.888	0.243	0.543
50	50	2.50	21	791	46	25	0.920	0.313	0.457
50	100	1.50	27	733	104	19	0.861	0.206	0.587
50	100	2.00	26	759	78	20	0.889	0.250	0.565
50	100	2.50	23	781	56	23	0.911	0.291	0.500
50	200	1.50	30	728	109	16	0.858	0.216	0.652
50	200	2.00	26	754	83	20	0.883	0.239	0.565
50	200	2.50	20	776	61	26	0.901	0.247	0.435
100	25	1.50	26	735	102	20	0.862	0.203	0.565
100	25	2.00	23	772	65	23	0.900	0.261	0.500
100	25	2.50	21	798	39	25	0.928	0.350	0.457
100	50	1.50	26	736	101	20	0.863	0.205	0.565
100	50	2.00	25	757	80	21	0.886	0.238	0.543
100	50	2.50	22	796	41	24	0.926	0.349	0.478
100	100	1.50	28	726	111	18	0.854	0.201	0.609
100	100	2.00	26	761	76	20	0.891	0.255	0.565
100	100	2.50	20	782	55	26	0.908	0.267	0.435
100	200	1.50	29	722	115	17	0.851	0.201	0.630
100	200	2.00	26	756	81	20	0.886	0.243	0.565
100	200	2.50	25	789	48	21	0.922	0.342	0.543
200	25	1.50	27	736	101	19	0.864	0.211	0.587
200	25	2.00	27	761	76	19	0.892	0.262	0.587
200	25	2.50	22	790	47	24	0.920	0.319	0.478
200	50	1.50	26	738	99	20	0.865	0.208	0.565
200	50	2.00	22	770	67	24	0.897	0.247	0.478
200	50	2.50	22	788	49	24	0.917	0.310	0.478
200	100	1.50	29	730	107	17	0.860	0.213	0.630
200	100	2.00	24	755	82	22	0.882	0.226	0.522
200	100	2.50	21	784	53	25	0.912	0.284	0.457
200	200	1.50	29	725	112	17	0.854	0.206	0.630
200	200	2.00	25	758	79	21	0.887	0.240	0.543
200	200	2.50	23	785	52	23	0.915	0.307	0.500

Table 9 presents the results of the *Random Forest Categorizer*, where P_1 denotes the number of random trees and P_2 denotes the number of random attributes.

Table 9. Random Forest Categorizer Results: First Review.

P_1	P_2	TN	FN	TP	FP	Accuracy	Precision	Recall
10	25	9	835	2	37	0.956	0.818	0.196
10	50	9	835	2	37	0.956	0.818	0.196
10	100	10	834	3	36	0.956	0.769	0.217
10	200	10	834	3	36	0.956	0.769	0.217
25	25	10	834	3	36	0.956	0.769	0.217
25	50	10	834	3	36	0.956	0.769	0.217
25	100	12	834	3	34	0.958	0.800	0.261
25	200	9	832	5	37	0.952	0.643	0.196
50	25	9	834	3	37	0.955	0.750	0.196
50	50	11	834	3	35	0.957	0.786	0.239
50	100	10	834	3	36	0.956	0.769	0.217
50	200	10	835	2	36	0.957	0.833	0.217
100	25	8	835	2	38	0.955	0.800	0.174
100	50	10	834	3	36	0.956	0.769	0.217
100	100	11	834	3	35	0.957	0.786	0.239
100	200	9	835	2	37	0.956	0.818	0.196
200	25	9	834	3	37	0.955	0.750	0.196
200	50	10	834	3	36	0.956	0.769	0.217
200	100	11	834	3	35	0.957	0.786	0.239
200	200	9	834	3	37	0.955	0.750	0.196

Table 10 presents the results of the *Naive Bayes Categorizer*, where P_1 denotes the minimum value to be regarded as equal to zero.

Table 10. Naïve Bayes Categorizer Results: First Review.

P_1	TN	FN	TP	FP	Accuracy	Precision	Recall
0.001	39	652	185	7	0.783	0.174	0.848
0.002	39	618	219	7	0.744	0.151	0.848
0.005	41	567	270	5	0.689	0.132	0.891
0.010	43	521	316	3	0.639	0.120	0.935

D.2 Ensemble Categorizers: First Review

Table 11 presents the top 200 results (sorted by recall) of the *Ensemble Categorizer*, where W_i denotes ensemble weight of the i^{th} individual categorizer. The individual categorizers (along with the specific categorizer parameters used) used in the ensemble are presented in Section 4.1.2.

Table 11. Ensemble Categorizer Results: First Review.

W_1	W_2	W_3	W_4	TP	TN	FP	FN	Accuracy	Precision	Recall
0.5	5	5	1.5	42	643	194	4	0.775764	0.177966	0.913043
1	5	5	2.5	42	642	195	4	0.774632	0.177215	0.913043
1	5	2.5	2.5	42	639	198	4	0.771234	0.175	0.913043
1	5	3	2.5	42	639	198	4	0.771234	0.175	0.913043
1	3	5	2.5	42	638	199	4	0.770102	0.174274	0.913043
0.5	5	3	1.5	42	637	200	4	0.768969	0.173554	0.913043
1	2	5	2.5	42	637	200	4	0.768969	0.173554	0.913043
1	2.5	5	2.5	42	637	200	4	0.768969	0.173554	0.913043
1	5	2	2.5	42	637	200	4	0.768969	0.173554	0.913043
0.5	5	2.5	1.5	42	636	201	4	0.767837	0.17284	0.913043
1	1.5	5	2.5	42	636	201	4	0.767837	0.17284	0.913043
0.5	3	5	1.5	42	635	202	4	0.766704	0.172131	0.913043
1	3	3	2.5	42	635	202	4	0.766704	0.172131	0.913043
1	5	1.5	2.5	42	635	202	4	0.766704	0.172131	0.913043
0.5	5	2	1.5	42	634	203	4	0.765572	0.171429	0.913043
1	1	5	2.5	42	634	203	4	0.765572	0.171429	0.913043
1	2.5	3	2.5	42	634	203	4	0.765572	0.171429	0.913043
1	3	2.5	2.5	42	634	203	4	0.765572	0.171429	0.913043
1	5	0.5	2.5	42	634	203	4	0.765572	0.171429	0.913043
1	5	1	2.5	42	634	203	4	0.765572	0.171429	0.913043
0.5	2.5	5	1.5	42	633	204	4	0.764439	0.170732	0.913043
1	0.5	5	2.5	42	632	205	4	0.763307	0.17004	0.913043
1	2	3	2.5	42	632	205	4	0.763307	0.17004	0.913043
1	2.5	2	2.5	42	632	205	4	0.763307	0.17004	0.913043
1	2.5	2.5	2.5	42	632	205	4	0.763307	0.17004	0.913043
1	3	1	2.5	42	632	205	4	0.763307	0.17004	0.913043
1	3	1.5	2.5	42	632	205	4	0.763307	0.17004	0.913043
1	3	2	2.5	42	632	205	4	0.763307	0.17004	0.913043
2	5	5	5	42	632	205	4	0.763307	0.17004	0.913043
0.5	2	5	1.5	42	631	206	4	0.762174	0.169355	0.913043
1	1.5	1.5	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	1.5	2	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	1.5	2.5	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	1.5	3	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	2	1.5	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	2	2	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	2	2.5	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	2.5	1	2.5	42	631	206	4	0.762174	0.169355	0.913043
1	2.5	1.5	2.5	42	631	206	4	0.762174	0.169355	0.913043
2	2.5	5	5	42	631	206	4	0.762174	0.169355	0.913043
2	3	3	5	42	631	206	4	0.762174	0.169355	0.913043
2	3	5	5	42	631	206	4	0.762174	0.169355	0.913043
2	5	2	5	42	631	206	4	0.762174	0.169355	0.913043
2	5	2.5	5	42	631	206	4	0.762174	0.169355	0.913043
2	5	3	5	42	631	206	4	0.762174	0.169355	0.913043
0.5	3	3	1.5	42	630	207	4	0.761042	0.168675	0.913043
1	1	3	2.5	42	630	207	4	0.761042	0.168675	0.913043
1	2	1	2.5	42	630	207	4	0.761042	0.168675	0.913043
1	3	0.5	2.5	42	630	207	4	0.761042	0.168675	0.913043
2	5	1.5	5	42	630	207	4	0.761042	0.168675	0.913043

W_1	W_2	W_3	W_4	TP	TN	FP	FN	Accuracy	Precision	Recall
0.5	1.5	5	1.5	42	629	208	4	0.759909	0.168	0.913043
0.5	5	1.5	1.5	42	629	208	4	0.759909	0.168	0.913043
1	1	2.5	2.5	42	629	208	4	0.759909	0.168	0.913043
1	2	0.5	2.5	42	629	208	4	0.759909	0.168	0.913043
1	2.5	0.5	2.5	42	629	208	4	0.759909	0.168	0.913043
2	2	5	5	42	629	208	4	0.759909	0.168	0.913043
2	2.5	3	5	42	629	208	4	0.759909	0.168	0.913043
2	3	2.5	5	42	629	208	4	0.759909	0.168	0.913043
2	5	0.5	5	42	629	208	4	0.759909	0.168	0.913043
2	5	1	5	42	629	208	4	0.759909	0.168	0.913043
0.5	1	5	1.5	42	628	209	4	0.758777	0.167331	0.913043
1	0.5	3	2.5	42	628	209	4	0.758777	0.167331	0.913043
1	1	2	2.5	42	628	209	4	0.758777	0.167331	0.913043
1	1.5	1	2.5	42	628	209	4	0.758777	0.167331	0.913043
2	1.5	5	5	42	628	209	4	0.758777	0.167331	0.913043
2	3	2	5	42	628	209	4	0.758777	0.167331	0.913043
1	0.5	2.5	2.5	42	627	210	4	0.757644	0.166667	0.913043
1	1.5	0.5	2.5	42	627	210	4	0.757644	0.166667	0.913043
2	1	5	5	42	627	210	4	0.757644	0.166667	0.913043
2	2.5	1.5	5	42	627	210	4	0.757644	0.166667	0.913043
2	2.5	2	5	42	627	210	4	0.757644	0.166667	0.913043
2	2.5	2.5	5	42	627	210	4	0.757644	0.166667	0.913043
2	3	0.5	5	42	627	210	4	0.757644	0.166667	0.913043
2	3	1	5	42	627	210	4	0.757644	0.166667	0.913043
2	3	1.5	5	42	627	210	4	0.757644	0.166667	0.913043
1	0.5	2	2.5	42	626	211	4	0.756512	0.166008	0.913043
1	1	1.5	2.5	42	626	211	4	0.756512	0.166008	0.913043
2	0.5	5	5	42	626	211	4	0.756512	0.166008	0.913043
2	2	3	5	42	626	211	4	0.756512	0.166008	0.913043
2	2.5	1	5	42	626	211	4	0.756512	0.166008	0.913043
0.5	2.5	3	1.5	42	625	212	4	0.755379	0.165354	0.913043
0.5	3	2.5	1.5	42	625	212	4	0.755379	0.165354	0.913043
0.5	5	1	1.5	42	625	212	4	0.755379	0.165354	0.913043
2	2	2.5	5	42	625	212	4	0.755379	0.165354	0.913043
2	2.5	0.5	5	42	625	212	4	0.755379	0.165354	0.913043
0.5	2	3	1.5	42	624	213	4	0.754247	0.164706	0.913043
0.5	2.5	2.5	1.5	42	624	213	4	0.754247	0.164706	0.913043
0.5	3	2	1.5	42	624	213	4	0.754247	0.164706	0.913043
0.5	5	0.5	1.5	42	624	213	4	0.754247	0.164706	0.913043
1	0.5	1	2.5	42	624	213	4	0.754247	0.164706	0.913043
1	0.5	1.5	2.5	42	624	213	4	0.754247	0.164706	0.913043
1	1	0.5	2.5	42	624	213	4	0.754247	0.164706	0.913043
1	1	1	2.5	42	624	213	4	0.754247	0.164706	0.913043
1	5	5	3	42	624	213	4	0.754247	0.164706	0.913043
2	1	1.5	5	42	624	213	4	0.754247	0.164706	0.913043
2	1	2	5	42	624	213	4	0.754247	0.164706	0.913043
2	1	2.5	5	42	624	213	4	0.754247	0.164706	0.913043
2	1	3	5	42	624	213	4	0.754247	0.164706	0.913043
2	1.5	0.5	5	42	624	213	4	0.754247	0.164706	0.913043
2	1.5	1	5	42	624	213	4	0.754247	0.164706	0.913043
2	1.5	1.5	5	42	624	213	4	0.754247	0.164706	0.913043
2	1.5	2	5	42	624	213	4	0.754247	0.164706	0.913043

W_1	W_2	W_3	W_4	TP	TN	FP	FN	Accuracy	Precision	Recall
2	1.5	2.5	5	42	624	213	4	0.754247	0.164706	0.913043
2	1.5	3	5	42	624	213	4	0.754247	0.164706	0.913043
2	2	0.5	5	42	624	213	4	0.754247	0.164706	0.913043
2	2	1	5	42	624	213	4	0.754247	0.164706	0.913043
2	2	1.5	5	42	624	213	4	0.754247	0.164706	0.913043
2	2	2	5	42	624	213	4	0.754247	0.164706	0.913043
0.5	0.5	5	1.5	42	623	214	4	0.753114	0.164063	0.913043
1	0.5	0.5	2.5	42	623	214	4	0.753114	0.164063	0.913043
2	0.5	1	5	42	623	214	4	0.753114	0.164063	0.913043
2	0.5	1.5	5	42	623	214	4	0.753114	0.164063	0.913043
2	0.5	2	5	42	623	214	4	0.753114	0.164063	0.913043
2	0.5	2.5	5	42	623	214	4	0.753114	0.164063	0.913043
2	0.5	3	5	42	623	214	4	0.753114	0.164063	0.913043
2	1	0.5	5	42	623	214	4	0.753114	0.164063	0.913043
2	1	1	5	42	623	214	4	0.753114	0.164063	0.913043
0.5	2	2.5	1.5	42	622	215	4	0.751982	0.163424	0.913043
0.5	1.5	3	1.5	42	621	216	4	0.750849	0.162791	0.913043
0.5	3	1.5	1.5	42	621	216	4	0.750849	0.162791	0.913043
2	0.5	0.5	5	42	621	216	4	0.750849	0.162791	0.913043
0.5	2.5	2	1.5	42	619	218	4	0.748584	0.161538	0.913043
0.5	3	1	1.5	42	619	218	4	0.748584	0.161538	0.913043
0.5	2	2	1.5	42	618	219	4	0.747452	0.16092	0.913043
0.5	2.5	1.5	1.5	42	618	219	4	0.747452	0.16092	0.913043
1	5	2.5	3	42	618	219	4	0.747452	0.16092	0.913043
1	5	3	3	42	618	219	4	0.747452	0.16092	0.913043
0.5	1	3	1.5	42	617	220	4	0.746319	0.160305	0.913043
0.5	3	0.5	1.5	42	617	220	4	0.746319	0.160305	0.913043
0.5	1.5	2.5	1.5	42	616	221	4	0.745187	0.159696	0.913043
0.5	2.5	1	1.5	42	616	221	4	0.745187	0.159696	0.913043
1	2.5	5	3	42	616	221	4	0.745187	0.159696	0.913043
1	3	5	3	42	616	221	4	0.745187	0.159696	0.913043
1	5	2	3	42	616	221	4	0.745187	0.159696	0.913043
0.5	0.5	3	1.5	42	614	223	4	0.742922	0.158491	0.913043
0.5	1	2.5	1.5	42	614	223	4	0.742922	0.158491	0.913043
0.5	2	1.5	1.5	42	614	223	4	0.742922	0.158491	0.913043
0.5	2.5	0.5	1.5	42	614	223	4	0.742922	0.158491	0.913043
1	2	5	3	42	614	223	4	0.742922	0.158491	0.913043
1	5	1	3	42	614	223	4	0.742922	0.158491	0.913043
1	5	1.5	3	42	614	223	4	0.742922	0.158491	0.913043
0.5	0.5	2.5	1.5	42	613	224	4	0.741789	0.157895	0.913043
0.5	2	1	1.5	42	613	224	4	0.741789	0.157895	0.913043
1	0.5	5	3	42	613	224	4	0.741789	0.157895	0.913043
1	1	5	3	42	613	224	4	0.741789	0.157895	0.913043
1	1.5	5	3	42	613	224	4	0.741789	0.157895	0.913043
0.5	1	2	1.5	42	612	225	4	0.740657	0.157303	0.913043
0.5	1.5	1.5	1.5	42	612	225	4	0.740657	0.157303	0.913043
0.5	1.5	2	1.5	42	612	225	4	0.740657	0.157303	0.913043
1	2.5	3	3	42	612	225	4	0.740657	0.157303	0.913043
1	3	2.5	3	42	612	225	4	0.740657	0.157303	0.913043
1	3	3	3	42	612	225	4	0.740657	0.157303	0.913043
1	5	0.5	3	42	612	225	4	0.740657	0.157303	0.913043
0.5	0.5	2	1.5	42	611	226	4	0.739524	0.156716	0.913043

W_1	W_2	W_3	W_4	TP	TN	FP	FN	Accuracy	Precision	Recall
0.5	1	1.5	1.5	42	611	226	4	0.739524	0.156716	0.913043
0.5	1.5	1	1.5	42	611	226	4	0.739524	0.156716	0.913043
1	1.5	3	3	42	611	226	4	0.739524	0.156716	0.913043
1	2	3	3	42	611	226	4	0.739524	0.156716	0.913043
1	2.5	2.5	3	42	611	226	4	0.739524	0.156716	0.913043
1	3	2	3	42	611	226	4	0.739524	0.156716	0.913043
0.5	2	0.5	1.5	42	610	227	4	0.738392	0.156134	0.913043
0.5	5	5	2	42	610	227	4	0.738392	0.156134	0.913043
1	2	2.5	3	42	610	227	4	0.738392	0.156134	0.913043
1	2.5	2	3	42	610	227	4	0.738392	0.156134	0.913043
0.5	0.5	1.5	1.5	42	609	228	4	0.737259	0.155556	0.913043
0.5	1.5	0.5	1.5	42	609	228	4	0.737259	0.155556	0.913043
1	1	3	3	42	609	228	4	0.737259	0.155556	0.913043
1	3	1	3	42	609	228	4	0.737259	0.155556	0.913043
1	3	1.5	3	42	609	228	4	0.737259	0.155556	0.913043
0.5	1	1	1.5	42	608	229	4	0.736127	0.154982	0.913043
1	1.5	2	3	42	608	229	4	0.736127	0.154982	0.913043
1	1.5	2.5	3	42	608	229	4	0.736127	0.154982	0.913043
1	2	1.5	3	42	608	229	4	0.736127	0.154982	0.913043
1	2	2	3	42	608	229	4	0.736127	0.154982	0.913043
1	2.5	1	3	42	608	229	4	0.736127	0.154982	0.913043
1	2.5	1.5	3	42	608	229	4	0.736127	0.154982	0.913043
1	3	0.5	3	42	608	229	4	0.736127	0.154982	0.913043
0.5	0.5	1	1.5	42	607	230	4	0.734994	0.154412	0.913043
0.5	1	0.5	1.5	42	607	230	4	0.734994	0.154412	0.913043
1	0.5	2.5	3	42	607	230	4	0.734994	0.154412	0.913043
1	0.5	3	3	42	607	230	4	0.734994	0.154412	0.913043
1	1	2	3	42	607	230	4	0.734994	0.154412	0.913043
1	1	2.5	3	42	607	230	4	0.734994	0.154412	0.913043
1	1.5	1.5	3	42	607	230	4	0.734994	0.154412	0.913043
1	2	1	3	42	607	230	4	0.734994	0.154412	0.913043
1	2.5	0.5	3	42	607	230	4	0.734994	0.154412	0.913043
1	0.5	2	3	42	605	232	4	0.732729	0.153285	0.913043
1	1	1.5	3	42	605	232	4	0.732729	0.153285	0.913043
1	1.5	1	3	42	605	232	4	0.732729	0.153285	0.913043
1	2	0.5	3	42	605	232	4	0.732729	0.153285	0.913043
0.5	0.5	0.5	1.5	42	604	233	4	0.731597	0.152727	0.913043
1	0.5	1.5	3	42	604	233	4	0.731597	0.152727	0.913043
1	1	1	3	42	604	233	4	0.731597	0.152727	0.913043
1	1.5	0.5	3	42	604	233	4	0.731597	0.152727	0.913043
1	0.5	1	3	42	603	234	4	0.730464	0.152174	0.913043
1	1	0.5	3	42	603	234	4	0.730464	0.152174	0.913043
1.5	5	5	5	42	603	234	4	0.730464	0.152174	0.913043
1	0.5	0.5	3	42	602	235	4	0.729332	0.151625	0.913043
1.5	2	5	5	42	601	236	4	0.728199	0.151079	0.913043
1.5	2.5	5	5	42	601	236	4	0.728199	0.151079	0.913043

D.3 Individual Categorizers: Second Review

Table 12 presents the results of the *Perceptron Categorizer*, where P_1 denotes the maximum number of iterations, P_2 denotes the margin of positive instances, and P_3 denotes the margin of negative instances.

Table 12. Perceptron Categorizer Results: Second Review.

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
100	1	1	1201	12168	1154	2553	0.78	0.51	0.32
100	1	10	707	12794	528	3047	0.79	0.57	0.19
100	1	50	537	13028	294	3217	0.79	0.65	0.14
100	1	100	290	13218	104	3464	0.79	0.74	0.08
100	10	1	2295	10358	2964	1459	0.74	0.44	0.61
100	10	10	1753	11610	1712	2001	0.78	0.51	0.47
100	10	50	712	12873	449	3042	0.80	0.61	0.19
100	10	100	372	13162	160	3382	0.79	0.70	0.10
100	50	1	3089	8075	5247	665	0.65	0.37	0.82
100	50	10	2708	9541	3781	1046	0.72	0.42	0.72
100	50	50	1483	12115	1207	2271	0.80	0.55	0.40
100	50	100	776	12905	417	2978	0.80	0.65	0.21
3000	50	0	1329	12196	1126	2425	0.79	0.54	0.35
3000	50	1	1324	12194	1128	2430	0.79	0.54	0.35
3000	50	10	1155	12397	925	2599	0.79	0.56	0.31
3000	100	0	1493	12049	1273	2261	0.79	0.54	0.40
3000	100	1	1472	12084	1238	2282	0.79	0.54	0.39
3000	100	10	1383	12241	1081	2371	0.80	0.56	0.37
3000	150	0	1582	12044	1278	2172	0.80	0.55	0.42
3000	150	1	1579	12043	1279	2175	0.80	0.55	0.42
3000	150	10	1515	12146	1176	2239	0.80	0.56	0.40
5000	50	0	1377	12196	1126	2377	0.79	0.55	0.37
5000	50	1	1362	12214	1108	2392	0.80	0.55	0.36
5000	50	10	1153	12494	828	2601	0.80	0.58	0.31
5000	100	0	1424	12177	1145	2330	0.80	0.55	0.38
5000	100	1	1410	12200	1122	2344	0.80	0.56	0.38
5000	100	10	1319	12310	1012	2435	0.80	0.57	0.35
5000	150	0	1506	12189	1133	2248	0.80	0.57	0.40
5000	150	1	1498	12194	1128	2256	0.80	0.57	0.40
5000	150	10	1405	12302	1020	2349	0.80	0.58	0.37
7000	50	0	1409	12190	1132	2345	0.80	0.55	0.38
7000	50	1	1406	12210	1112	2348	0.80	0.56	0.37
7000	50	10	1248	12404	918	2506	0.80	0.58	0.33
7000	100	0	1438	12179	1143	2316	0.80	0.56	0.38
7000	100	1	1438	12177	1145	2316	0.80	0.56	0.38
7000	100	10	1339	12312	1010	2415	0.80	0.57	0.36
7000	150	0	1528	12150	1172	2226	0.80	0.57	0.41
7000	150	1	1520	12157	1165	2234	0.80	0.57	0.40
7000	150	10	1456	12281	1041	2298	0.80	0.58	0.39

Table 13 presents the results of the *Balanced Perceptron Categorizer*, where P_1 denotes the maximum number of iterations, P_2 denotes the margin of positive instances, P_3 denotes the margin of negative instances, P_4 denotes the number of balanced learners, and P_5 denotes the majority class percentage used to train each learner.

Table 13. Balanced Perceptron Categorizer Results: Second Review.

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
50	25	5	5	2.0	10	13313	9	3744	0.78	0.53	0.00
50	25	5	5	2.5	6	13313	9	3748	0.78	0.40	0.00
50	25	5	5	3.0	5	13317	5	3749	0.78	0.50	0.00
50	25	5	10	2.0	9	13314	8	3745	0.78	0.53	0.00
50	25	5	10	2.5	9	13316	6	3745	0.78	0.60	0.00
50	25	5	10	3.0	6	13316	6	3748	0.78	0.50	0.00
50	25	5	15	2.0	8	13315	7	3746	0.78	0.53	0.00
50	25	5	15	2.5	6	13316	6	3748	0.78	0.50	0.00
50	25	5	15	3.0	6	13317	5	3748	0.78	0.55	0.00
50	25	10	5	2.0	3	13317	5	3751	0.78	0.38	0.00
50	25	10	5	2.5	5	13317	5	3749	0.78	0.50	0.00
50	25	10	5	3.0	2	13317	5	3752	0.78	0.29	0.00
50	25	10	10	2.0	4	13317	5	3750	0.78	0.44	0.00
50	25	10	10	2.5	3	13318	4	3751	0.78	0.43	0.00
50	25	10	10	3.0	4	13317	5	3750	0.78	0.44	0.00
50	25	10	15	2.0	4	13317	5	3750	0.78	0.44	0.00
50	25	10	15	2.5	5	13318	4	3749	0.78	0.56	0.00
50	25	10	15	3.0	4	13318	4	3750	0.78	0.50	0.00
50	25	15	5	2.0	3	13318	4	3751	0.78	0.43	0.00
50	25	15	5	2.5	3	13321	1	3751	0.78	0.75	0.00
50	25	15	5	3.0	3	13319	3	3751	0.78	0.50	0.00
50	25	15	10	2.0	3	13318	4	3751	0.78	0.43	0.00
50	25	15	10	2.5	3	13318	4	3751	0.78	0.43	0.00
50	25	15	10	3.0	1	13320	2	3753	0.78	0.33	0.00
50	25	15	15	2.0	3	13317	5	3751	0.78	0.38	0.00
50	25	15	15	2.5	2	13321	1	3752	0.78	0.67	0.00
50	25	15	15	3.0	2	13319	3	3752	0.78	0.40	0.00
50	50	5	5	2.0	35	13304	18	3719	0.78	0.66	0.01
50	50	5	5	2.5	30	13308	14	3724	0.78	0.68	0.01
50	50	5	5	3.0	24	13310	12	3730	0.78	0.67	0.01
50	50	5	10	2.0	43	13301	21	3711	0.78	0.67	0.01
50	50	5	10	2.5	29	13308	14	3725	0.78	0.67	0.01
50	50	5	10	3.0	30	13307	15	3724	0.78	0.67	0.01
50	50	5	15	2.0	38	13303	19	3716	0.78	0.67	0.01
50	50	5	15	2.5	24	13307	15	3730	0.78	0.62	0.01
50	50	5	15	3.0	24	13307	15	3730	0.78	0.62	0.01
50	50	10	5	2.0	33	13307	15	3721	0.78	0.69	0.01
50	50	10	5	2.5	21	13311	11	3733	0.78	0.66	0.01
50	50	10	5	3.0	21	13311	11	3733	0.78	0.66	0.01
50	50	10	10	2.0	28	13308	14	3726	0.78	0.67	0.01
50	50	10	10	2.5	21	13311	11	3733	0.78	0.66	0.01
50	50	10	10	3.0	21	13310	12	3733	0.78	0.64	0.01
50	50	10	15	2.0	28	13307	15	3726	0.78	0.65	0.01
50	50	10	15	2.5	20	13310	12	3734	0.78	0.63	0.01

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
50	50	10	15	3.0	20	13311	11	3734	0.78	0.65	0.01
50	50	15	5	2.0	25	13310	12	3729	0.78	0.68	0.01
50	50	15	5	2.5	17	13315	7	3737	0.78	0.71	0.00
50	50	15	5	3.0	16	13317	5	3738	0.78	0.76	0.00
50	50	15	10	2.0	22	13311	11	3732	0.78	0.67	0.01
50	50	15	10	2.5	17	13314	8	3737	0.78	0.68	0.00
50	50	15	10	3.0	18	13312	10	3736	0.78	0.64	0.00
50	50	15	15	2.0	22	13310	12	3732	0.78	0.65	0.01
50	50	15	15	2.5	18	13315	7	3736	0.78	0.72	0.00
50	50	15	15	3.0	20	13314	8	3734	0.78	0.71	0.01
50	100	5	5	2.0	101	13286	36	3653	0.78	0.74	0.03
50	100	5	5	2.5	79	13298	24	3675	0.78	0.77	0.02
50	100	5	5	3.0	69	13298	24	3685	0.78	0.74	0.02
50	100	5	10	2.0	102	13284	38	3652	0.78	0.73	0.03
50	100	5	10	2.5	72	13299	23	3682	0.78	0.76	0.02
50	100	5	10	3.0	68	13299	23	3686	0.78	0.75	0.02
50	100	5	15	2.0	100	13282	40	3654	0.78	0.71	0.03
50	100	5	15	2.5	76	13297	25	3678	0.78	0.75	0.02
50	100	5	15	3.0	76	13296	26	3678	0.78	0.75	0.02
50	100	10	5	2.0	91	13291	31	3663	0.78	0.75	0.02
50	100	10	5	2.5	69	13298	24	3685	0.78	0.74	0.02
50	100	10	5	3.0	61	13301	21	3693	0.78	0.74	0.02
50	100	10	10	2.0	89	13291	31	3665	0.78	0.74	0.02
50	100	10	10	2.5	63	13299	23	3691	0.78	0.73	0.02
50	100	10	10	3.0	65	13301	21	3689	0.78	0.76	0.02
50	100	10	15	2.0	89	13292	30	3665	0.78	0.75	0.02
50	100	10	15	2.5	68	13299	23	3686	0.78	0.75	0.02
50	100	10	15	3.0	63	13300	22	3691	0.78	0.74	0.02
50	100	15	5	2.0	76	13297	25	3678	0.78	0.75	0.02
50	100	15	5	2.5	60	13301	21	3694	0.78	0.74	0.02
50	100	15	5	3.0	51	13303	19	3703	0.78	0.73	0.01
50	100	15	10	2.0	87	13296	26	3667	0.78	0.77	0.02
50	100	15	10	2.5	56	13302	20	3698	0.78	0.74	0.01
50	100	15	10	3.0	58	13302	20	3696	0.78	0.74	0.02
50	100	15	15	2.0	75	13297	25	3679	0.78	0.75	0.02
50	100	15	15	2.5	55	13301	21	3699	0.78	0.72	0.01
50	100	15	15	3.0	54	13302	20	3700	0.78	0.73	0.01
100	25	5	5	2.0	28	13306	16	3726	0.78	0.64	0.01
100	25	5	5	2.5	19	13306	16	3735	0.78	0.54	0.01
100	25	5	5	3.0	25	13309	13	3729	0.78	0.66	0.01
100	25	5	10	2.0	23	13308	14	3731	0.78	0.62	0.01
100	25	5	10	2.5	17	13309	13	3737	0.78	0.57	0.00
100	25	5	10	3.0	19	13308	14	3735	0.78	0.58	0.01
100	25	5	15	2.0	29	13306	16	3725	0.78	0.64	0.01
100	25	5	15	2.5	21	13309	13	3733	0.78	0.62	0.01
100	25	5	15	3.0	23	13309	13	3731	0.78	0.64	0.01
100	25	10	5	2.0	21	13311	11	3733	0.78	0.66	0.01
100	25	10	5	2.5	16	13312	10	3738	0.78	0.62	0.00
100	25	10	5	3.0	15	13315	7	3739	0.78	0.68	0.00
100	25	10	10	2.0	20	13310	12	3734	0.78	0.63	0.01
100	25	10	10	2.5	15	13311	11	3739	0.78	0.58	0.00
100	25	10	10	3.0	15	13313	9	3739	0.78	0.63	0.00

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
100	25	10	15	2.0	21	13310	12	3733	0.78	0.64	0.01
100	25	10	15	2.5	14	13312	10	3740	0.78	0.58	0.00
100	25	10	15	3.0	15	13314	8	3739	0.78	0.65	0.00
100	25	15	5	2.0	12	13313	9	3742	0.78	0.57	0.00
100	25	15	5	2.5	7	13314	8	3747	0.78	0.47	0.00
100	25	15	5	3.0	8	13315	7	3746	0.78	0.53	0.00
100	25	15	10	2.0	20	13311	11	3734	0.78	0.65	0.01
100	25	15	10	2.5	9	13315	7	3745	0.78	0.56	0.00
100	25	15	10	3.0	9	13314	8	3745	0.78	0.53	0.00
100	25	15	15	2.0	14	13312	10	3740	0.78	0.58	0.00
100	25	15	15	2.5	11	13313	9	3743	0.78	0.55	0.00
100	25	15	15	3.0	9	13313	9	3745	0.78	0.50	0.00
100	50	5	5	2.0	73	13295	27	3681	0.78	0.73	0.02
100	50	5	5	2.5	58	13297	25	3696	0.78	0.70	0.02
100	50	5	5	3.0	48	13300	22	3706	0.78	0.69	0.01
100	50	5	10	2.0	73	13294	28	3681	0.78	0.72	0.02
100	50	5	10	2.5	51	13298	24	3703	0.78	0.68	0.01
100	50	5	10	3.0	54	13298	24	3700	0.78	0.69	0.01
100	50	5	15	2.0	65	13297	25	3689	0.78	0.72	0.02
100	50	5	15	2.5	54	13298	24	3700	0.78	0.69	0.01
100	50	5	15	3.0	48	13299	23	3706	0.78	0.68	0.01
100	50	10	5	2.0	61	13298	24	3693	0.78	0.72	0.02
100	50	10	5	2.5	47	13302	20	3707	0.78	0.70	0.01
100	50	10	5	3.0	42	13306	16	3712	0.78	0.72	0.01
100	50	10	10	2.0	56	13300	22	3698	0.78	0.72	0.01
100	50	10	10	2.5	49	13300	22	3705	0.78	0.69	0.01
100	50	10	10	3.0	39	13304	18	3715	0.78	0.68	0.01
100	50	10	15	2.0	55	13296	26	3699	0.78	0.68	0.01
100	50	10	15	2.5	46	13302	20	3708	0.78	0.70	0.01
100	50	10	15	3.0	44	13299	23	3710	0.78	0.66	0.01
100	50	15	5	2.0	46	13301	21	3708	0.78	0.69	0.01
100	50	15	5	2.5	38	13303	19	3716	0.78	0.67	0.01
100	50	15	5	3.0	35	13307	15	3719	0.78	0.70	0.01
100	50	15	10	2.0	50	13301	21	3704	0.78	0.70	0.01
100	50	15	10	2.5	35	13307	15	3719	0.78	0.70	0.01
100	50	15	10	3.0	37	13303	19	3717	0.78	0.66	0.01
100	50	15	15	2.0	49	13301	21	3705	0.78	0.70	0.01
100	50	15	15	2.5	36	13304	18	3718	0.78	0.67	0.01
100	50	15	15	3.0	36	13304	18	3718	0.78	0.67	0.01
100	100	5	5	2.0	125	13281	41	3629	0.79	0.75	0.03
100	100	5	5	2.5	98	13289	33	3656	0.78	0.75	0.03
100	100	5	5	3.0	98	13292	30	3656	0.78	0.77	0.03
100	100	5	10	2.0	132	13278	44	3622	0.79	0.75	0.04
100	100	5	10	2.5	102	13285	37	3652	0.78	0.73	0.03
100	100	5	10	3.0	99	13291	31	3655	0.78	0.76	0.03
100	100	5	15	2.0	129	13277	45	3625	0.79	0.74	0.03
100	100	5	15	2.5	101	13287	35	3653	0.78	0.74	0.03
100	100	5	15	3.0	100	13291	31	3654	0.78	0.76	0.03
100	100	10	5	2.0	115	13278	44	3639	0.78	0.72	0.03
100	100	10	5	2.5	90	13289	33	3664	0.78	0.73	0.02
100	100	10	5	3.0	98	13292	30	3656	0.78	0.77	0.03
100	100	10	10	2.0	115	13282	40	3639	0.78	0.74	0.03

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
100	100	10	10	2.5	87	13291	31	3667	0.78	0.74	0.02
100	100	10	10	3.0	92	13293	29	3662	0.78	0.76	0.02
100	100	10	15	2.0	120	13282	40	3634	0.78	0.75	0.03
100	100	10	15	2.5	93	13291	31	3661	0.78	0.75	0.02
100	100	10	15	3.0	94	13292	30	3660	0.78	0.76	0.03
100	100	15	5	2.0	110	13284	38	3644	0.78	0.74	0.03
100	100	15	5	2.5	88	13294	28	3666	0.78	0.76	0.02
100	100	15	5	3.0	82	13296	26	3672	0.78	0.76	0.02
100	100	15	10	2.0	110	13285	37	3644	0.78	0.75	0.03
100	100	15	10	2.5	88	13293	29	3666	0.78	0.75	0.02
100	100	15	10	3.0	81	13295	27	3673	0.78	0.75	0.02
100	100	15	15	2.0	110	13284	38	3644	0.78	0.74	0.03
100	100	15	15	2.5	87	13293	29	3667	0.78	0.75	0.02
100	100	15	15	3.0	85	13293	29	3669	0.78	0.75	0.02
200	25	5	5	2.0	50	13297	25	3704	0.78	0.67	0.01
200	25	5	5	2.5	40	13298	24	3714	0.78	0.63	0.01
200	25	5	5	3.0	45	13298	24	3709	0.78	0.65	0.01
200	25	5	10	2.0	55	13296	26	3699	0.78	0.68	0.01
200	25	5	10	2.5	43	13300	22	3711	0.78	0.66	0.01
200	25	5	10	3.0	43	13298	24	3711	0.78	0.64	0.01
200	25	5	15	2.0	48	13296	26	3706	0.78	0.65	0.01
200	25	5	15	2.5	39	13297	25	3715	0.78	0.61	0.01
200	25	5	15	3.0	42	13298	24	3712	0.78	0.64	0.01
200	25	10	5	2.0	39	13300	22	3715	0.78	0.64	0.01
200	25	10	5	2.5	36	13300	22	3718	0.78	0.62	0.01
200	25	10	5	3.0	34	13302	20	3720	0.78	0.63	0.01
200	25	10	10	2.0	42	13302	20	3712	0.78	0.68	0.01
200	25	10	10	2.5	33	13303	19	3721	0.78	0.63	0.01
200	25	10	10	3.0	32	13302	20	3722	0.78	0.62	0.01
200	25	10	15	2.0	41	13300	22	3713	0.78	0.65	0.01
200	25	10	15	2.5	37	13304	18	3717	0.78	0.67	0.01
200	25	10	15	3.0	34	13304	18	3720	0.78	0.65	0.01
200	25	15	5	2.0	39	13303	19	3715	0.78	0.67	0.01
200	25	15	5	2.5	32	13305	17	3722	0.78	0.65	0.01
200	25	15	5	3.0	34	13307	15	3720	0.78	0.69	0.01
200	25	15	10	2.0	35	13303	19	3719	0.78	0.65	0.01
200	25	15	10	2.5	26	13305	17	3728	0.78	0.60	0.01
200	25	15	10	3.0	31	13306	16	3723	0.78	0.66	0.01
200	25	15	15	2.0	35	13302	20	3719	0.78	0.64	0.01
200	25	15	15	2.5	31	13306	16	3723	0.78	0.66	0.01
200	25	15	15	3.0	32	13305	17	3722	0.78	0.65	0.01
200	50	5	5	2.0	101	13283	39	3653	0.78	0.72	0.03
200	50	5	5	2.5	82	13286	36	3672	0.78	0.69	0.02
200	50	5	5	3.0	79	13289	33	3675	0.78	0.71	0.02
200	50	5	10	2.0	107	13281	41	3647	0.78	0.72	0.03
200	50	5	10	2.5	87	13289	33	3667	0.78	0.73	0.02
200	50	5	10	3.0	84	13289	33	3670	0.78	0.72	0.02
200	50	5	15	2.0	101	13282	40	3653	0.78	0.72	0.03
200	50	5	15	2.5	85	13290	32	3669	0.78	0.73	0.02
200	50	5	15	3.0	76	13289	33	3678	0.78	0.70	0.02
200	50	10	5	2.0	86	13284	38	3668	0.78	0.69	0.02
200	50	10	5	2.5	76	13295	27	3678	0.78	0.74	0.02

P_1	P_2	P_3	P_4	P_5	TN	FN	TP	FP	Accuracy	Precision	Recall
200	50	10	5	3.0	71	13293	29	3683	0.78	0.71	0.02
200	50	10	10	2.0	81	13289	33	3673	0.78	0.71	0.02
200	50	10	10	2.5	67	13294	28	3687	0.78	0.71	0.02
200	50	10	10	3.0	66	13293	29	3688	0.78	0.69	0.02
200	50	10	15	2.0	84	13289	33	3670	0.78	0.72	0.02
200	50	10	15	2.5	70	13292	30	3684	0.78	0.70	0.02
200	50	10	15	3.0	71	13293	29	3683	0.78	0.71	0.02
200	50	15	5	2.0	79	13293	29	3675	0.78	0.73	0.02
200	50	15	5	2.5	53	13295	27	3701	0.78	0.66	0.01
200	50	15	5	3.0	61	13298	24	3693	0.78	0.72	0.02
200	50	15	10	2.0	78	13292	30	3676	0.78	0.72	0.02
200	50	15	10	2.5	60	13297	25	3694	0.78	0.71	0.02
200	50	15	10	3.0	60	13301	21	3694	0.78	0.74	0.02
200	50	15	15	2.0	81	13293	29	3673	0.78	0.74	0.02
200	50	15	15	2.5	59	13295	27	3695	0.78	0.69	0.02
200	50	15	15	3.0	62	13294	28	3692	0.78	0.69	0.02
200	100	5	5	2.0	172	13257	65	3582	0.79	0.73	0.05
200	100	5	5	2.5	129	13280	42	3625	0.79	0.75	0.03
200	100	5	5	3.0	118	13279	43	3636	0.78	0.73	0.03
200	100	5	10	2.0	178	13266	56	3576	0.79	0.76	0.05
200	100	5	10	2.5	122	13282	40	3632	0.78	0.75	0.03
200	100	5	10	3.0	121	13282	40	3633	0.78	0.75	0.03
200	100	5	15	2.0	171	13266	56	3583	0.79	0.75	0.05
200	100	5	15	2.5	119	13282	40	3635	0.78	0.75	0.03
200	100	5	15	3.0	115	13285	37	3639	0.78	0.76	0.03
200	100	10	5	2.0	150	13268	54	3604	0.79	0.74	0.04
200	100	10	5	2.5	120	13282	40	3634	0.78	0.75	0.03
200	100	10	5	3.0	116	13282	40	3638	0.78	0.74	0.03
200	100	10	10	2.0	161	13270	52	3593	0.79	0.76	0.04
200	100	10	10	2.5	107	13286	36	3647	0.78	0.75	0.03
200	100	10	10	3.0	109	13287	35	3645	0.78	0.76	0.03
200	100	10	15	2.0	154	13268	54	3600	0.79	0.74	0.04
200	100	10	15	2.5	114	13283	39	3640	0.78	0.75	0.03
200	100	10	15	3.0	111	13287	35	3643	0.78	0.76	0.03
200	100	15	5	2.0	149	13271	51	3605	0.79	0.75	0.04
200	100	15	5	2.5	109	13289	33	3645	0.78	0.77	0.03
200	100	15	5	3.0	109	13286	36	3645	0.78	0.75	0.03
200	100	15	10	2.0	144	13277	45	3610	0.79	0.76	0.04
200	100	15	10	2.5	105	13290	32	3649	0.78	0.77	0.03
200	100	15	10	3.0	102	13287	35	3652	0.78	0.74	0.03
200	100	15	15	2.0	128	13275	47	3626	0.78	0.73	0.03
200	100	15	15	2.5	111	13289	33	3643	0.78	0.77	0.03
200	100	15	15	3.0	104	13289	33	3650	0.78	0.76	0.03

Table 14 presents the results of the *Balanced Random Forest 1 Categorizer*, where P_1 denotes the number of random trees, P_2 denotes the number of random attributes, and P_3 denotes the majority class percentage used to train each tree.

Table 14. Balanced Random Forest 1 Categorizer Results: Second Review.

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
100	50	1.0	3010	9261	4061	744	0.72	0.43	0.80
100	50	1.5	2356	11152	2170	1398	0.79	0.52	0.63
100	50	2.0	1755	12069	1253	1999	0.81	0.58	0.47
100	100	1.0	3043	9319	4003	711	0.72	0.43	0.81
100	100	1.5	2333	11272	2050	1421	0.80	0.53	0.62
100	100	2.0	1924	12030	1292	1830	0.82	0.60	0.51
100	200	1.0	3083	9262	4060	671	0.72	0.43	0.82
100	200	1.5	2387	11177	2145	1367	0.79	0.53	0.64
100	200	2.0	1987	11954	1368	1767	0.82	0.59	0.53
200	50	1.0	2975	9375	3947	779	0.72	0.43	0.79
200	50	1.5	2308	11214	2108	1446	0.79	0.52	0.61
200	50	2.0	1781	12104	1218	1973	0.81	0.59	0.47
200	100	1.0	3074	9348	3974	680	0.73	0.44	0.82
200	100	1.5	2341	11280	2042	1413	0.80	0.53	0.62
200	100	2.0	1906	12030	1292	1848	0.82	0.60	0.51
200	200	1.0	3098	9329	3993	656	0.73	0.44	0.83
200	200	1.5	2373	11186	2136	1381	0.79	0.53	0.63
200	200	2.0	2028	11951	1371	1726	0.82	0.60	0.54
300	50	1.0	2972	9343	3979	782	0.72	0.43	0.79
300	50	1.5	2318	11216	2106	1436	0.79	0.52	0.62
300	50	2.0	1762	12127	1195	1992	0.81	0.60	0.47
300	100	1.0	3051	9301	4021	703	0.72	0.43	0.81
300	100	1.5	2329	11248	2074	1425	0.80	0.53	0.62
300	100	2.0	1880	12049	1273	1874	0.82	0.60	0.50

Table 15 presents the results of the *Balanced Random Forest 1 Categorizer*, where P_1 denotes the number of random trees, P_2 denotes the number of random attributes, and P_3 denotes the majority class percentage used to train each tree.

Table 15. Balanced Random Forest 2 Categorizer Results: Second Review.

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
5	50	2.0	1689	11983	1339	2065	0.80	0.56	0.45
5	50	2.5	1351	12341	981	2403	0.80	0.58	0.36
5	50	3.0	944	12822	500	2810	0.81	0.65	0.25
5	100	2.0	1842	11840	1482	1912	0.80	0.55	0.49
5	100	2.5	1396	12368	954	2358	0.81	0.59	0.37
5	100	3.0	1227	12667	655	2527	0.81	0.65	0.33
5	200	2.0	1958	11786	1536	1796	0.80	0.56	0.52
5	200	2.5	1560	12174	1148	2194	0.80	0.58	0.42
5	200	3.0	1406	12549	773	2348	0.82	0.65	0.37
10	50	2.0	1726	12069	1253	2028	0.81	0.58	0.46
10	50	2.5	1245	12573	749	2509	0.81	0.62	0.33

P_1	P_2	P_3	TN	FN	TP	FP	Accuracy	Precision	Recall
10	50	3.0	978	12858	464	2776	0.81	0.68	0.26
10	100	2.0	1807	11989	1333	1947	0.81	0.58	0.48
10	100	2.5	1534	12384	938	2220	0.82	0.62	0.41
10	100	3.0	1210	12723	599	2544	0.82	0.67	0.32
10	200	2.0	1911	11905	1417	1843	0.81	0.57	0.51
10	200	2.5	1611	12312	1010	2143	0.82	0.61	0.43
10	200	3.0	1437	12611	711	2317	0.82	0.67	0.38
20	50	2.0	1746	12004	1318	2008	0.81	0.57	0.47
20	50	2.5	1247	12558	764	2507	0.81	0.62	0.33
20	50	3.0	1082	12816	506	2672	0.81	0.68	0.29
20	100	2.0	1853	11959	1363	1901	0.81	0.58	0.49
20	100	2.5	1478	12402	920	2276	0.81	0.62	0.39
20	100	3.0	1274	12726	596	2480	0.82	0.68	0.34
20	200	2.0	1966	11958	1364	1788	0.82	0.59	0.52
20	200	2.5	1624	12296	1026	2130	0.82	0.61	0.43
20	200	3.0	1429	12621	701	2325	0.82	0.67	0.38

Table 16 presents the results of the *Naive Bayes Categorizer*, where P_1 denotes the minimum value to be regarded as equal to zero.

Table 16. Naïve Bayes Categorizer Results: Second Review.

P_1	TN	FN	TP	FP	Accuracy	Precision	Recall
0.000100	1519	11751	1571	2235	0.78	0.49	0.40
0.000010	1515	11791	1531	2239	0.78	0.50	0.40
0.000001	1505	11829	1493	2249	0.78	0.50	0.40

This page intentionally left blank.

APPENDIX E. THE NAÏVE BAYES CLASSIFIER

E.1 Detailed Description

Independence among the features describing a domain (of interest) dominates the theory of model formation in naïve Bayes classifiers (Barber, 2005; Eibe and Bouckaert, 2006; Eyherandy, *et al.*, 2003; McCallum and Nigam, 1998; Rennie, 2001; Shen and Jiang, 2003). Examples in a domain are described by their feature values

$$(f_1, f_2, \ldots, f_n).$$

The probability that a particular set of feature values represents a specific class, c, is

$$p(C = c \mid F_1 = f_1, F_2 = f_2, \ldots, F_n = f_n) = \frac{p(C = c)\, p(F_1 = f_1, F_2 = f_2, \ldots, F_n = f_n \mid C = c)}{P(F_1 = f_1, F_2 = f_2, \ldots, F_n = f_n)}.$$

In English, this says that the a posteriori probability of class c given the feature values $(f_1, f_2, \ldots, f_n)$ is the a priori probability of class c times the likelihood of $(f_1, f_2, \ldots, f_n)$ given class c divided by the evidence for $(f_1, f_2, \ldots, f_n)$. Note that the denominator is independent of the classes, and so when a comparison is needed (between two or more classes) this value can be ignored (since it will be the same for all classes). The prior probability for class c is estimated as the frequency of class c over the total number of training samples. Since all features are assumed to be independent from one another (this is the naïve assumption), then the likelihood is

$$p(C = c \mid F_1 = f_1, F_2 = f_2, \ldots F_n = f_n) = \frac{1}{k}\, p(C = c) \prod_{i=1}^{n} p(F_i = f_i \mid C = c),$$

where k is the probability of the feature values.

There are two potential problems with using naïve Bayes classifiers. First, it is possible for the likelihood for a particular feature given a specific class to be exactly zero (i.e., $p(F_i = f_i \mid C = c) = 0$). In this case, the likelihood product would also be zero. This problem can be dealt with by ensuring that none of the feature/class likelihood estimates is allowed to be exactly zero (i.e., $p(F_i = f_i \mid C = c) \geq \varepsilon > 0, \forall i$ for some small constant ε). A second problem is that the product of feature likelihoods can very rapidly approach zero (or become so small that it is zero to the precision that a computer can maintain). One method to address this second issue is to use the logarithm of the maximum likelihood (Barber, 2005). The naïve Bayes with log-likelihood classifier is then

$$\begin{aligned} \textit{classify}(f_1, f_2, \ldots, f_n) &= \underset{C=c}{\operatorname{argmax}} \ln(p(C=c)\prod_{i=1}^{n} p(F_i = f_i \mid C=c)) \\ &= \underset{C=c}{\operatorname{argmax}} [\ln(p(C=c)) + \sum_{i=1}^{n} \ln(p(F_i = f_i \mid C=c))] \end{aligned}$$

Benefits of the naïve Bayes classifier include that it does not require a large amount of data to train, and training can be accomplished in a very short amount of time.

Determining a confidence value for naïve Bayes classification can be problematic since the likelihood values can approach 0 and/or 1 very fast (especially when there are many features, i.e., when n is large). The current confidence value that we are using for the naïve Bayes classifier is

$$\frac{1}{1 - \max_{c}[\ln(p(C=c)) - \sum_{i=1}^{n} \ln(p(F_i = f_i \mid C=c)]}$$

With this confidence value, the farther a sample's classification is from "perfect" the larger the denominator will be in the confidence value (approaching 0), meaning less confidence in the classification. Note that this confidence measure will vary between 0 and 1, inclusively.

APPENDIX F. FUZZY ARTMAP NETWORK

The Fuzzy ARTMAP neural network is composed of two Fuzzy ART neural networks connected through a map field module (Carpenter, *et al.*, 1992). Each Fuzzy ART network performs unsupervised learning, and the map field links (or associates) a left-side Fuzzy ART category template (hyper-rectangle) with a right-side Fuzzy ART template (see Figure 9). The map field associating the Fuzzy ART modules facilitates supervised learning.

F.1 Fuzzy ART Unsupervised Learning Algorithm

For a given training sample, the Fuzzy ART learning algorithm has three stages (Carpenter, *et al.*, 1991). First, the input is complement coded. Then the best matching F2 node is found for the complement coded input data. Note that the F2 node thus found might be the uncommitted node, and initially, a Fuzzy ART neural network architecture ha s only the single uncommitted node available for learning. Finally, the best matching F2 node is allowed to learn the new data point.

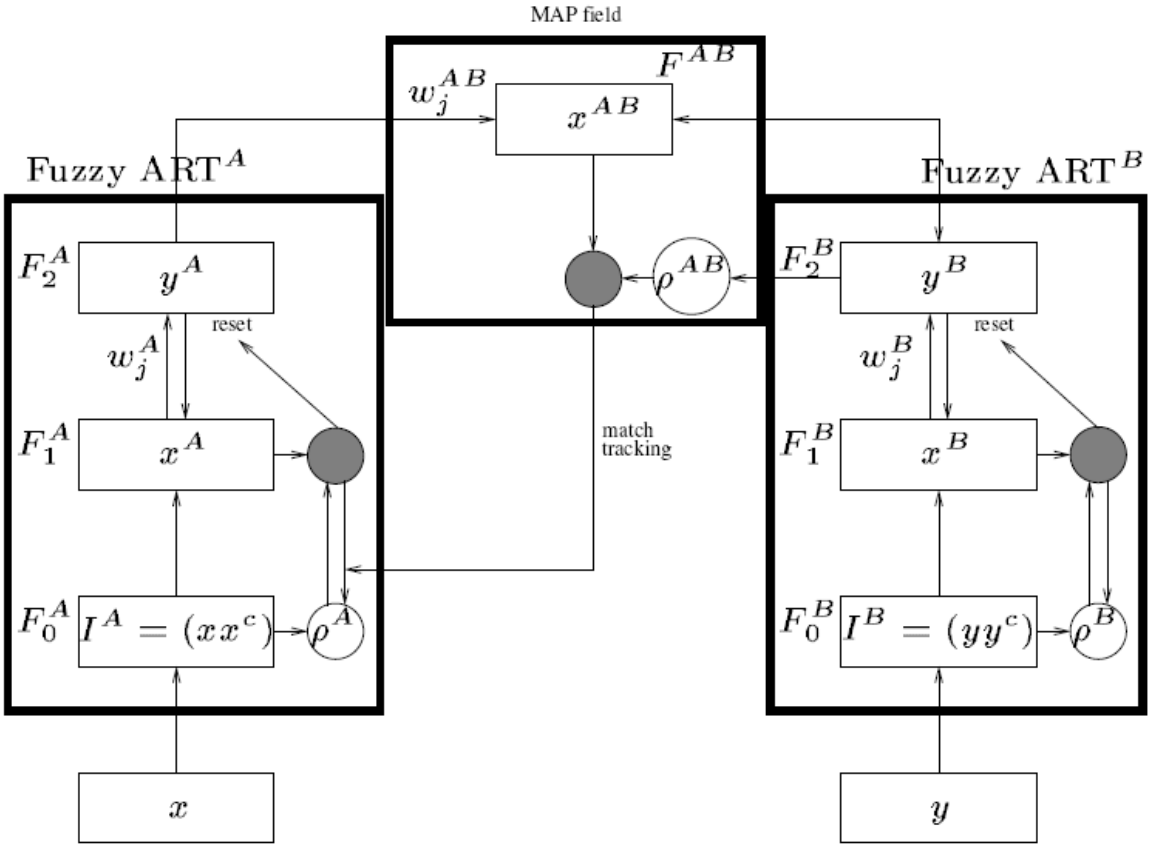

Figure 9. Fuzzy ARTMAP Architecture.

The mathematical formula used by Fuzzy ART to find the best matching category template during cluster formation is

$$I = (xx^c),$$
$$J = \underset{0 \le j \le N}{\operatorname{argmax}} T_j(I),$$
$$T_j(I) = \begin{cases} \dfrac{|I \wedge w_j|}{\alpha + |w_j|}, & if \dfrac{|I \wedge w_j|}{|I|} \ge \rho \\ 0, & otherwise \end{cases}.$$

The parameter, α, called the choice parameter, is usually a small positive quantity, $\wedge$ is the element-wise vector min operator, and $|\cdot|$ is the L1-norm of a vector. The best matching F2 node, *J*, from the choice competition must satisfy the vigilance criterion

$$\frac{|I \wedge w_j|}{|I|} \ge \rho.$$

The vigilance parameter, ρ, is a user-supplied input between zero and one. Note that at least one F2 node, the uncommitted node, will always satisfy the vigilance criterion. The maximum choice F2 template satisfying the vigilance criterion is allowed to learn the input vector, a condition called resonance. Ties between F2 nodes with the same choice value are broken by assigning an index to all committed F2 nodes, and choosing the node with the lowest index value in a tie. Initially all template weights are set to one, and learning proceeds as follows

$$w_J^{(new)} = \beta(I \wedge w_J^{(old)}) + (1-\beta)w_J^{(old)}$$

where β is the learning parameter. In most learning cases, including SCF, $\beta = 1$, which is a special case called fast learning.

F.2 Fuzzy ARTMAP Supervised Learning Algorithm

The Fuzzy ARTMAP neural network is composed of two Fuzzy ART networks (left and right) connected through a map field. The left-side network is given domain data (*n*-dimensional feature data), and the right-side network is given the label associated with the domain data. In classification, each left-side cluster (hyper-rectangle) formed is linked with only one right-side label cluster facilitating for supervised learning.

Initially, the Fuzzy ARTMAP map field weights are set to 1 (this is a temporary state that indicates that all possible link associations are available). When an association is learned in the map field (between a left-side cluster, J, and right-side cluster, K), the map field weights are set as

$$w_{Jk}^{MAP} = \begin{cases} 1 & k = K \\ 0 & k \ne K \end{cases}$$

At this point, the link between *J* (left) and *K* (right) is fixed, and thus left-side cluster, *J*, will only attempt to learn training data associated with label *K*. The Fuzzy ARTMAP neural network

ensures a many to one mapping between left and right side Fuzzy ART templates through the use of match tracking and lateral reset (Carpenter, *et al.*, 1992).

The vigilance parameter for the left-side Fuzzy ART network is initially set to zero, but during learning, it is allowed to increase as necessary to facilitate the lateral reset mechanism. After learning occurs (for each training sample), this vigilance value is returned to its initial state. The right-side vigilance parameter is usually set to one to facilitate crisp classification (i.e., no clustering of labels). There is also a vigilance parameter in the map field which is also set to one for crisp classification. These parameters need not be tuned any further for nominal Fuzzy ARTMAP operation.

The confidence value for Fuzzy ARTMAP categorization in SCF is computed as a normalized, inverted distance from the F2 template centroid. A Fuzzy ARTMAP F2 template contains a vector which represents a hyper-rectangle in the *n*-dimensional feature space

$$w_j = (w_1, w_2, \ldots, w_{2n}) = (p_1, p_2, \ldots, p_n, q_1, q_2, \ldots, q_n).$$

The first half of the template vector represents the lower left corner of the hyper-rectangle, (p_1, p_2, …, p_n), and the complement of the second half, (1-q_1, 1-q_2, …, 1-q_n) represents the upper right corner. Therefore the template centroid can be computed as

$$c_j = (c_1, c_s, \ldots, c_n) = \left(\frac{p_1 + 1 - q_1}{2}, \frac{p_2 + 1 - q_2}{2}, \ldots, \frac{p_n + 1 - q_n}{2}\right).$$

Given the input sample, x, and winning F2 node, w_j, the confidence is computed as

$$\frac{1}{1 + d(c_j, x)},$$

Where $d(c_j, x)$ is the distance from x to the template centroid. Note that the distance measure could be either Euclidean or the L1-norm (used in Fuzzy ART). In the SCF, we used the Euclidean distance measure for computing Fuzzy ARTMAP confidence. Note that the confidence value varies between 0 and 1, inclusive, and a value closer to 0 indicates less confidence in the categorization.

DISTRIBUTION

1	MS0721	Steve Roehrig	6300
1	MS1011	Wendy Shaneyfelt	6343
1	MS1011	Steve Verzi	6343
1	MS1188	Justin Basilico	6341
1	MS1188	Gerry Sleefe	6340
1	MS1188	John Wagner	6341
1	MS1318	Daniel Dunlavy	1415
1	MS1399	Gordon Appel	6033
1	MS0899	Technical Library	9536 (electronic copy)


Sandia National Laboratories