Effect of Imbalanced Data on Document Classification Algorithms

Amrita Paul

A thesis submitted to
Auckland University of Technology
for the fulfilment of the requirements for the degree
of
Master of Computer and Information Science (MCIS)

2014

School of Computer and Mathematical Sciences

Abstract

Text classification is the task of assigning predefined categories to free text documents. Due to the ever-increasing amount of electronic documents, digital libraries and web resources, document classification is critical in higher level document processing tasks such as information extraction, named entity recognition and event modelling. Text categorization is considered to be challenging because of the large number of features in a typical text document. In spite of this, various categorization algorithms have reached accuracies in the vicinity of 90%. It has generally been found that probability based algorithms perform better on Natural Language Processing tasks compared to other types of algorithms. This is in addition to probabilistic algorithms being highly extensible.

In this thesis paper, a tool called MALLET (MAchine Learning for LanguagE Toolkit) was used to perform document classification using a set of probabilistic algorithms to determine the effect of imbalanced data on the performance of these algorithms when compared to balanced data. The data used for the research was taken from Reuters Corpus (RCV1) which contains categorized newspaper articles. Although the corpus contains many fine levels of categorization, this research used four upper level topic codes which were further organized into binary categories of a document belonging to a category or out of it. The documents were then converted into a form acceptable to MALLET and tested for categorization with the chosen algorithms.

The algorithms used for the research were Naïve Bayes, Balanced Winnow and three variations of Max Ent, namely Max Ent, Max Ent L1 and MC Max Ent. It was firstly found that these probability based algorithms performed marginally better than other algorithms reported in previous works on similar genre of input data. However, a significant finding from the research was that the algorithms performed similarly or in some cases even better, for imbalanced data compared to balanced data. This was due to the vocabulary properties of the documents used for training and asserts the resilience of the probability based algorithms for text categorization.

Contents

List of Figures

List of Tables

Attestation of Authorship

I hereby declare that this thesis is my own work. To the best of my knowledge and belief, it contains no materials previously written or published by any other person, except for the authors defined in the acknowledgements. I also declare that this work has not been submitted to any other institution or university for the award of any other degree or diploma in any university or other institution of higher learning.

The thesis work was conducted from July 2012 to February 2014 under the supervision of Dr. Parma Nand at Auckland University of Technology.

Amrita Paul
Auckland, New Zealand
February 2014

Acknowledgements

I wish to specially acknowledge individuals who have helped directly or indirectly for the accomplishment of my thesis.

First of all, I would like to thank my primary supervisor, Dr. Parma Nand. He has been my advisor and proof reader throughout the journey towards completion of my thesis. I would like to thank him for his wisdom and tireless assistance. I have learnt enormously from his supervision, intuition and intensive discussions that we have had, both on the research area as well as the general research. I would like to thank Dr. Russel Pears for being my secondary supervisor and guiding me with his comments improving my writing.

Next, I would like to thank my husband, Sujoy Biswas and my only 7 year old daughter Emili Biswas for enduring my not being able to cook and do other household chores on a regular basis and also for not giving much time to Emili. Moreover I would like to thank Sujoy for cooking dinner for us often after his work.

I would like to thank the faculty of the School of Computing and Mathematical Sciences of AUT. I would like to thank all the staff members of the department for providing words of encouragement and responses to information required.

The completion of this thesis would not have been possible had it not been for the direct and indirect contribution of my friends and family. As it is hard to express every name individually, I would like to express my gratitude to each of them for their well-wishes and steadfast trust in me.

Lastly, I would like to thank my extended family from Kolkata in India, especially my parents and my brother for inspiring and encouraging me over the telephone till the end of my thesis.

Chapter 1. *Introduction*

1.1 Background

Currently, an enormous amount of information is allied with Web technology and the internet. Due to the increasing number of electronic resources both in structured and unstructured forms, such as news articles, biological databases, online forums, digital libraries and blog repositories, the task of text categorization is essential where data needs to be automatically classified into categories before other Natural Language Processing (NLP) tasks can be applied. The emergence of the World Wide Web (WWW), led to the handling of a large amount of electronic text data as highlighted by Krishnalal, Rengarajan, & Srinivasagan (2010). Document classification has gained an increasing amount of interest due to the ever increasing need to extract information from free texts for purposes such as business intelligence, security, and social needs such as to combat cyber bullying. Document classification or document categorization is the task of assigning a document to one more classes or categories and is mostly applicable in library science, information science or computer science. To retrieve valuable information from documents, the text has to be categorized. To automatically categorize documents is a challenge, due to the large number of features in a typical document. The task of mechanically arranging a set of documents into categories from a predefined set is known as text categorization or text classification (TC). Sebastiani (2005) introduced many applications of TC such as mechanized indexing of scientific articles according to pre-set dictionaries of procedural terms; categorization of academic papers into technical domains and sub-domains; automated population of hierarchical catalogues of Web resources; and patient reports in health-care organizations often indexed on multiple aspects, e.g. using taxonomies of disease categories, types of surgical procedures, insurance reimbursement codes.

Some more applications of TC include automatic indexing for Boolean information retrieval system, document organization, text filtering, word sense disambiguation, automated survey coding and authorship attribution, genre classification, spam filtering and so on. The application of text categorization involves the retrieval of information from some form of archive as Reuters Corpus which contains a large

amount data in a structured form such as Corporate or Industrial data, Economics data, Market data and Government or Social data.

Within the application of text categorization, its key success lies with the involvement of machine learning techniques. According to Sebastiani (2005), the task of assigning documents to pre-defined categories falls at the junction of information retrieval (IR) and machine learning (ML). The text classification accuracy with the combination of IR and ML rivals human trained professionals. Automated text categorization generated on a rule-base requires human expertise such as indexing expertise which is expensive as well as time consuming as proposed by Sebastiani (2005). This type of text categorization is interesting because it saves organizations from manually organising the documents into their respective categories. However the rule based categorization systems are not extendible or require extensive human effort for extension, which has given rise to the increasing prominence of probability based categorization techniques.

Text categorization can be done using various types of text based features used with machine learning techniques. Some examples of features that have been used are cue phrases, word frequency, and discourse segmentation. Various combinations of these features are then used in a machine learning algorithm in order to train the algorithm to classify previously unseen documents into predefined categories. Examples of machine learning algorithms that have been used in many researches are nearest neighbour classifiers, Bayesian classifiers, decision trees, support vector machines, neural networks, maximum entropy modelling, perceptron, Latent Semantic Analysis and Genetic Algorithms.

Research on text categorization has been conducted since the early 2000s and now there are quite effective algorithms being used in several applications to perform the text categorization task successfully. The recent classifying algorithms found from McCallum (2013) used for text classifications are Ada Boost, Balanced Winnow, C45, Winnow, Max Ent and MC Max Ent which have achieved accuracies up to 0.990.

Some of the tools that have been used recently to perform text classification are Apache Mahout 0.8, MALLET 2.0.7 and Alchemy API. Apache Mahout 0.8 includes complementary Naïve Bayes Classifier, Random Forest Decision Tree based Classifier, Winnow and Perceptron (*The Apache Software Foundation*, 2014).

MALLET 2.0.7 includes Naïve Bayes, Maximum Entropy, and Decision Trees for converting text to features, Hidden Markov Models, Maximum Entropy Markov Models and Conditional Random Fields for sequence tagging for applications such as the extraction of text from named-entity (McCallum, 2013).

Alchemy API includes sophisticated statistical algorithms and NLP (Natural Language Processing Technology) to analyse information. It can categorize HTML and web based content (*Alchemy API, Inc*, 2013).

Text categorization deals with large amounts of data. The most frequent problem with text categorization is that the data may be skewed or imbalanced. Data imbalance is the existence of a large number of files in one category as compared to others. This is common because typical classification tasks involve classifying objects in a category, or out of it. In this situation, the number of objects in the category are fewer than the number not in the category. Research on text classification from imbalanced data has drawn attention in the field of academia and industry. Imbalanced data can lead to poor classification accuracy or even misclassification. There are many algorithms found to be successful in terms of achieving good classification accuracy when dealing with imbalanced data. For example, AdaBoost has been considered to be a successful algorithm to improve the classification accuracy, as reported in Sun, Kamel, Wong, & Wang (2007). The imbalanced data problem is common in any type of dataset and for this reason; specific Learning models are made with specific algorithms to achieve higher accuracies. Many researchers have attempted to solve the issue of reduced accuracy for imbalanced data with varying amount of success.

This thesis examines the effect of data imbalance on a selection of probabilistic classification algorithms. Specifically, the objective was to examine the level of imbalance in the data that would start to cause a drop in the accuracy of classifications. The data for the research was taken from the RCV1 corpus which contains approximately 1,611,205 newspaper articles classified into 3 sets of categories, namely Topics, Industries and Regions, according to Lewis, Yang, Rose, & Li (2004). Some of the articles in the corpus were classified in more than one category. Only mutually exclusive articles from the corpus data were chosen to be used for this research. These articles were organized into binary categories and tested on Naïve Bayes, Balanced Winnow, Max Ent, MC Max Ent and Max EntL1 implemented in MALLET (McCallum, 2013). It was found that the imbalance of data did not have any significant

effect on the classification performance of the selected algorithms. For example, the Max Ent L1 algorithm gave an accuracy of 96.7 % on balanced data for 6060 documents, while the accuracy for the same algorithm was 98.4 % for 5050 documents with an imbalance of 5:6. The results of this thesis show the resilience of the algorithms to imbalanced data, whereas the use of neural network models reported in Olson (2005) show the results deteriorate for imbalanced data. According to Yanling Li, Sun, & Zhu (2010), *kNN* and SVM algorithms are non-adaptive to imbalanced data and resulted lower accuracies in Krishnalal et al. (2010) when compared to the results of this thesis.

1.2 Outline of Thesis

Chapter 1 discusses the thesis objective and background. Chapter 2 describes the previous works on document categorization. Chapter 3 describes the methodology including input data, tools and algorithms used for the experiments as well as the experimental setup. Chapter 4 describes the experimental results. Chapter 5 discusses the overall results followed by a conclusion and future works in Chapter 6.

Chapter 2. *Literature Review*

This chapter starts by providing an overview of the generic algorithms that have been effectively used for the purpose of classification. Section 2.2 is a review of works that have adapted these algorithms for classifying these documents. This is followed by section 2.3 which reviews works that have focussed on classification of documents in the imbalanced data paradigm. This section highlights the gaps in the works on imbalanced data, which forms the basis of the research for this thesis.

In text categorization a variety of classification techniques are involved, for example, decision trees, maximum entropy modelling, perceptron and *k*-nearest neighbour. These algorithms along with some probabilistic algorithms are briefly described in section 2.1. The probabilistic algorithms that have been used in the research are described in further detail in section 3.3 of Chapter 3.

2.1 An overview of Classification Algorithms

Decision trees are the first class of the classification models which make decisions to assign documents to a category. A document is classified by starting at the top node; questions are tested; branching is done to the appropriate node and then the process is repeated until the leaf node is reached. Decision trees are computationally more expensive than other classifiers such as Naïve Bayes, linear regression and logistic regression. If the classification problem is simple, then the simpler method is preferable. Decision trees can split training sets into smaller subsets. As a result, correct generalization becomes harder if the data is not enough for reliable prediction whereas incorrect generalization becomes easier if smaller sets have accidental regularities that do not generalize. The advantage of decision tree is that they can be understood easily. With the help of a decision tree, the path from the root to the leaf node can be traced for articles which provide an understanding of how the document was classified by looking at questions at each of the previous nodes. Although decision trees are not widely used in the real world due to their computational costs, they are still important in terms of understanding the foundations of classifications, as well as their usefulness in small scale applications (Manning & Schiitze, 1999).

Perceptron is a gradient descent, hill climbing algorithm which is an important class of iterative learning algorithms in neural networks. In gradient descent, a function of the data is optimized that computes the goodness of the criterion such as likelihood. In each step, the derivative of the function is computed and the parameters of the model are changed in the direction of the steepest gradient, which may either be steepest ascent or descent depending on the optimality function. The idea can be sound because the direction of the steepest gradient is the direction where the most improvement can be expected in the goodness criterion (Manning & Schiitze, 1999).

The **Nearest Neighbour** classification rule is remarkably simple. The new object can be classified when a similar object is found in the training set. The main idea is that if there is an identical article (or at least one with the same representation), then the decision is to allocate the same category. If there is no identical article, then the most similar one is the best choice. The *k*- nearest neighbour (*kNN*) classification is a generalization of the nearest neighbour. *kNN* is used instead of one nearest neighbour when creating a base for decisions (Manning & Schiitze, 1999).

Naïve Bayes, also known as multinomial Naïve Bayes or multinomial NB, is considered as to be the first supervised probabilistic learning method (Manning, Raghavan, & Schiitze, 2008). Naïve Bayes uses the Bayes Theorem, which is a formula that computes a probability by counting the combination and frequency of values in the old data.

Balanced Winnow is a probabilistic machine learning technique that can identify linear classifiers from categorized examples. This algorithm can scale high dimensional data by maintaining two sets of weights or hyper planes to label new examples as optimistic or negative. This algorithm can be used in an online learning setting where the classification and learning phase are not distinguished properly. This online learning setting is referred to as the online machine learning model of induction, which can learn one instance at a time. The objective of online learning is to predict labels from instances.

Maximum Entropy modelling or Max Ent is another probabilistic machine learning technique. It is one of the frameworks for assimilating information from many heterogeneous information sources for the purpose of classification. Having a large number of features in the data can be a problem for several classification algorithms. These features in the data can be quite complex, and require the experimenter to have

prior knowledge of the type of features which are essential for classification. Each feature corresponds to a constraint on the maximum entropy model and the sum of the constraints is used to compute the model. In Maximum Entropy modelling, feature selection and training are usually integrated. The expectation of each feature, based on the training set, is computed for a given set of features. The constraint gets defined by each feature where the empirical expectation is same as the expectation the feature is having in the final maximum entropy model. The attempt is to find the maximum entropy distribution on the probability distributions that obey the constraints and also to find the one with the highest entropy. Maximum entropy distribution is a unique model which includes algorithms and generative iterative scaling methods. Generalized iterative scaling is the procedure for finding maximum entropy distribution. This procedure is expensive due to slow convergence. In the case of binary classification, the log linear model with linear separator is more powerful than Naïve Bayes or linear regression where the classifiers can be trained more proficiently. Generalized iterative scaling considers dependence between features in contrast to Naïve Bayes and other linear classifiers. If feature dependence is not a problem, then Naïve Bayes can be better than Maximum Entropy modelling (Manning & Schiitze, 1999). Max Ent has wide applications in areas such as Spatial Physics, Computer Vision, and Natural Language Processing.

Some of the probabilistic algorithms listed above, were used as the basis of experiments carried out in this thesis, and are further discussed in Chapter 3. These probabilistic algorithms were considered as they are computationally faster, easier and less expensive than other algorithms being used in research.

2.2 Related Works on Text Classification

There are many algorithms and techniques that can be used for classification of electronic documents. For example, Khan, Baharudin, Lee, & Khan (2010) describe supervised classification techniques, where the training documents are already classified into pre-defined categories. This paper highlights some of the challenges in text representation for Machine Learning (ML) techniques. Khan et al. (2010) presented some of the new hybrid approaches for classifying documents. For instance, a hybrid approach with the combination of the Naïve Bayes vectorizer and the SVM classifier shows improved classification accuracy over the pure Naïve Bayes classifier. Another hybrid combination referred to in the paper was Naive Bayes with Self Optimizing Map (SOM). Among the algorithms mentioned by Khan et al. (2010) were SVM, Naïve

Bayes and *kNN*. Combinations of these algorithms with appropriate feature selection techniques were found to give excellent results. Bi, Bell, Wang, Guo, & Greer (2004) investigated combining algorithms by combining four different classification methods using Dempster's rule of combination for text categorization. The classification methods included Support Vector Machine (SVM), *kNN* (nearest neighbours), *kNN* model-based approach (*kNNM*) and Rocchio methods. First an approach of effectively combining different classification methods was presented. This approach led to the application of the classification methods to the benchmark data collection of 20 newsgroups, both individually as well as in combination. The experimental result showed that the best performance achieved was on 10 groups of benchmark data by combining different classifiers. The classification accuracy achieved in combination with the classification methods mentioned was 91.07%. This was an average 2.68% better than the best individual method of SVM. The advantage of classification methods is that their strengths can be expressed by avoiding their individual weaknesses; as a result the classification accuracy can be improved. Bao & Ishii (2002) also considered the combination of multiple classifiers as an effective technique for improving classification accuracy. Cho & Lee (2003) suggested text classification for real application as most of the investigations in text classification are not conducted on real data. The authors suggested two methods of combining various types of neural networks to improve classification performance. Khan et al. (2010) demonstrated a feature selection process which can be used to improve classification accuracy and can even speed up the training process. For example, Bratu, Muresan, & Potolea (2008) considered feature selection, which mainly selects a subset of relevant features for making strong learning models as an effective procedure. The relevant feature depends on the target analysis, which can be improving classification accuracy, reducing computation effort to increase speed, avoiding expensive features, or improving model interpretability. The paper analysed a wrapper approach for feature selection to enhance the classification accuracy. The wrapper consisted of: a generation procedure such as forward greedy stepwise, backward greedy stepwise, forward best-first search, and backward best-first search; an evaluation function containing inducers such as C4.5, Naïve Bayes and Adaboost; and a validation procedure. The main focus was to analyse the behaviour of the most encouraging blend of search methods for generation and inducers for evaluation and validation so that the accuracy increment is assured. Like learning algorithms, there are no universally best feature selection methods, but there can be a method which produces highest accuracy enhancements on any dataset over

other methods. The experimental results proved that the wrapper approach always improves the classifier performance. The inducer which achieved highest accuracy may even show its high performance after feature selection (first or second best performance) on the assessed datasets. Several comprehensive studies have been carried out on feature selection algorithms. Dash & Liu (1997) classified feature selection algorithms utilizing two criteria like generation procedure and evaluation function.

Based on machine learning techniques namely the Hidden Markov Model (HMM) and Support Vector Machine (SVM), a system for online news classifications was proposed by Krishnalal et al. (2010). Keywords were extracted from online newspaper content by an intelligent system and classified into predefined categories using three different stages. The three stages were:

- Text pre-processing
- HMM based feature Extraction
- Classification using SVM

The data consisted of 1200 news articles taken for experimentation from newspapers such as The New Indian Express, Times of India, Business Line, and The Economic Times. The news articles were distributed among three major categories; sports, finance and politics. In the experiment, 800 texts were used as the training set and 400 texts were used as the testing set. Features were extracted from texts of every class using HMM. Multiclass SVMs were trained to provide a separating decision hyper plane that maximizes the limit of the classified categories. From the politics category, 118 out of 130 were classified correctly and from the finance news category, 158 out of 164 were classified correctly. When misclassified news articles were analysed, ambiguities in features were discovered. For example, the article headline "The terror attack on Sri Lankan cricket players at Lahore" contains features that can mislead the classifiers used to classify the article as sports news, whereas it actually belongs to the international news category. The classification accuracy using *kNN* was found to be 83.25%, 80.22%, and 82.26 % for sports, finance and politics respectively. The classification accuracy using SVM was found to be 87.67%, 82.57%, and 86.55% for sports, finance and politics respectively. For the same dataset the classification accuracy using the combined model HMM-SVM was found to be 92.45% for sports, 96.34% for finance, 90.76% for politics. The paper reports that the highest classification accuracy is

obtained using the HMM-SVM hybrid model compared to the individual algorithms. The training computational time was reduced significantly due to pre-processing in the training data. In this paper, the novel approach was basically to combine two powerful algorithms such as HMM and SVM and then compare the combined performance with the SVM and HMM models individually.

Hassan, Rafi, & Shaikh, (2012) carried out many experiments to perform text categorization by incorporating background knowledge to the document through knowledge repositories namely Wikipedia, Wikitology, Word Net, and the Open Project Directory (OPD). This paper mainly compares the performance of SVM (Support Vector Machines) and NB (Naïve Bayes) under text enrichment using Wikitology. The dataset comprised of 20 news groups which were well-balanced. There were 20 categories in the dataset, with 1000 documents in each category. The experimental baseline was setup by removing stop words and delimiters and the dataset was stemmed using Porter Stemmer. The experimental results were evaluated using micro-average and macro-average F-measures through SVM and NB. The data was evaluated using 10-fold cross validation with paired t-test. Different combined enrichment techniques were used for text document representation. The baselines A1 to A5 for background knowledge were used. A1 and A3 indicate top 20 titles and their related categories. A2 and A4 indicate top 5 titles and their related categories. Applying text classification techniques SVM and NB on experiments A1 to A5, the macro-average F-measures, micro-average F-measures and improvement or declination of the experiments were calculated. Using SVM, a micro-average F-measure of 0.919 and a macro-average F-measure of 0.920, improvements of +5.88% and +6.36% respectively were measured. Using NB, a micro-average F-measure of 0.881 and a macro-average F-measure of 0.877 were measured as well. The improvement was achieved on baseline A4 and on enrichment techniques E1, E4 and E5. The worst results achieved using the SVM classifier, showed a micro-average F-measure of 0.770 with a declination of -11.29%, and a macro-average F-measure of 0.757 with a declination of -12.49%. The worst result achieved using NB classifier, showed a micro-average F-measure of 0.685 with a declination of -1.15%, and a macro-average F-measure of 0.676 with a declination of -0.73%. An improvement of 0.649 to 0.714 was evaluated from dataset Reuters-21578 and that of 0.667 to 0.719 from 20 News Group dataset. The authors suggested pursuing similar experiments using *k*-Nearest Neighbour, Matrix Regression and Decision Trees on datasets Reuters-21578, RCV1, OSHUMED and Movies.

Ting, Ip, & Tsang (July 2011) compared the performance of Naïve Bayes with other common classifying algorithms namely decision tree (DT), Support Vector Machines (SVM), and neural networks in terms of computational efficiency and accuracy. A specified experimental dataset was considered where the datasets were to be classified into four correctly classified categories like business, politics, sports and travel. Each category had about 1000 documents to generate the classification model and was evaluated by splitting 4000 documents into two datasets as training and testing. The training set consisted of 1200 documents, or 30 % of the total, whereas the testing set consisted of 2800 documents, or 70 % of the total. The data had 1311 attributes considering numerical values and 1 solution attribute considering nominal values. The objective of the experimental evaluation was twofold. Firstly, the experiment investigated whether the pre-processing phase was useful to infer improved classification accuracy and performance when compared with the non-pre-processed data. Secondly, it compared the classification accuracy when different classifiers were applied. The model built was Naïve Bayes classifier based developed in WEKA. The non-pre-processed data yielded better classification accuracy of 96.9% compared to the pre-processed data yielding 95.5% accuracies. Initially Cfs Subset Evaluator and Rank search were used for feature selection where 75 attributes were used as input. Then an improved feature selection technique known as Chi-square was used. To determine whether Naïve Bayes is the best classifier, it is compared to other classifiers, namely SVM using SMO function in WEKA; Neural Network (the lazy IBk), and DT (the tree "J48"). In this case, the pre-processed data with 90 attributes were used for evaluation. Naïve Bayes was found to be best classifier where time consumed for building was only 0.19 seconds.

Kamruzzaman (2012) proposed a new algorithm for text classification based on artificial intelligence techniques where new documents were used for training. The main aim of this paper was to obtain classification accuracy with less training data and less computational time. The proposed algorithm was the implementation of hybrid methods such as Naïve Bayes classifier, the association rule and genetic algorithm. Instead of words, the word relation, that is the association rules from these words in the documents, was used. These words were used to develop a feature set from pre-classified text documents. Using Naïve Bayes concept features were developed and then the concept of genetic algorithm was applied for final classification. Abstracts from different papers were used for experimentation. The five different classes of

papers of physics (PH), chemistry (CH), algorithm (ALG), artificial intelligence (AI) and educational engineering (EDE) were chosen. The total number of abstracts used were 296, out of which 104 were physics, 88 were chemistry, 27 were algorithm, 62 were AI and 15 were EDE. After pre-processing of the data, 10 % of the total data was used as training data which showed an unsatisfactory accuracy of 31%. The training data was then increased to 20%, which showed developing accuracy. On increasing the percentage of training data further, the accuracy became much more desirable. The training data was increased up to 55%, which provided an accuracy of 68%. The best accuracy achieved was 80% when the training data was 50%.

Many different algorithms were compared with the proposed algorithm. The techniques compared with the proposed algorithm were Naïve Bayes (NB), Association Rule and Naïve Bayes, Association Rule Based Decision Tree, Genetic Algorithm and Conference Management. Using NB, 70% of the training data yielded an accuracy of 74%. Equally, different highest accuracies were obtained at different percentages of training data. The proposed algorithm achieved the highest accuracies of 80% using 50% of the training data. The difference between the proposed algorithm and the Association Rule Based Decision Tree was that the former algorithm used less training data (50%) to yield 80% of accuracy whereas the latter used 76% of training data to yield 87% of accuracy. So, the requirement of less training data and less computational time could be achieved through the proposed hybrid algorithm. But the author encouraged further investigation using his proposed algorithm with a large training dataset.

Al-Mubaid & Umair (2006) used the distributional clustering method on information bottleneck (IB) to represent the documents efficiently. A learning logic two-class classification technique called Lsquare was used for training text classifiers. The training data was viewed as logic formulas which were represented as vector entries of [0, ±1]. Each vector represented one document and each [0, ±1] entry in the vector represented a term / feature in the document. The text documents were represented in a suitable form for this learning algorithm. Each document was converted to a vector where the entries of the vector were word-cluster features. The clustering algorithm was applied on each dataset to obtain effective representation of input documents. The input words were distributed over all the categories of the whole dataset. The number of clusters required was pre-specified to the algorithm. The concept of hard clustering was used where one cluster was assigned to each input word. For all the experiments in

this paper, 120 clusters were chosen for extensive testing that contained almost all the words in the input data excluding the stop words. The clusters were effective in increasing the computational performance and computational time. Different numbers of stop words were removed from different datasets. The clustering approach gave a word stemming effect, which is the reduction of infecting words from their roots. Moreover, words with the same stem were residing in the same cluster. The training and testing documents were represented as vectors of dimension 'k' where 'k' is the number of clusters in the evaluations. As Lsquare accepts only binary data ([0, ±1] vectors), features were assigned binary values of +1 to indicate the presence of any words and -1 to indicate that none of the words of the cluster occurred in the document. For SVM, each feature in the vector space signified the words' cluster counts (i.e. the number of words of the cluster present in the document). The TC (Text categorization) approach implemented in this paper were fully implemented and evaluated with extensive experimentation. The same experimental settings were applied on different datasets such as WebKB, 20 Newsgroup and Reuters-21578 with the application of SVM and the classification accuracy and F1 results were compared. The experimental methods were applied on a limited amount of labelled training data. SVM was applied with other areas as distributional clustering and yielded a good result. Lsquare was also applied with distributional clustering. The author says that data imbalance problems in text categorization become a real issue. The imbalance problem arises when there are non-category documents in large numbers of categories. The non-category documents are considered as negative sets whereas the category documents are positive sets. Different techniques are suggested in Ruiz & Srinivasan (1999) to address the data imbalance problem. In this paper, equal amounts of positive and negative training documents were selected and the experiment was repeated 10 times using randomly selected positive and negative training documents. Four categories - 'course', 'faculty', 'student' and 'project' - were considered from the different datasets mentioned above. Using Lsquare, the macro-average and micro-average accuracy was 93.63% and 96.55% respectively. Equally, different macro-average and micro-average accuracy were found using SVM. The F1 macro-average and F1 micro-average was 89.18% and 92.81% respectively using Lsquare. Equally, the F1 macro-average and F1 micro-average was 83.80% and 88.25% respectively using SVM. The paper stated that the proposed method (Lsquare) outperformed SVM using training-testing sets of almost all aforementioned datasets. As a result, the experimental result showed that the new TC

method (distributional clustering method using Lsquare) was mainly useful on limited training data.

Harrang, El-Qawasmeh, & Pichappan (2009) presented the result of classifying Arabic text documents using the decision tree algorithm. Experiments were performed on two self-collected Arabic data corpora identified as scientific and literacy corpora. The results show that the hybrid approach of Document Frequency Thresholding, using the information gain criterion of the decision tree algorithm, is the desirable feature selection criterion. The scientific corpus consisted of 373 documents distributed over 8 categories taken from Arabian Scientific encyclopaedia "*Do you know*". The literacy corpus consisted of 453 documents distributed over 14 categories taken from the Prophetic encyclopaedia. This paper mainly focused on applying machine learning techniques that are commonly applicable to text categorization for Arabic languages. The experiment conducted for this paper mainly sought to evaluate the performance of the Decision Trees classification algorithm (ID3) on classifying Arabic Text using the above mentioned Arabic corpora. One-third of the Arabic data set was used for training the classifier and two-thirds of the dataset was used for testing. Texts of the datasets were represented in the form of vectors having 'm' number of elements that denotes the number of features mostly the text words. As high dimension input space in vectors normally affects the efficiency of the classification algorithms, term selection techniques were used that select a subset of terms from a super vector terms that signifying the meaning of documents. Several values were tested for term frequency (TF), document frequency (DF) and combined frequency (TF/DF). The literature corpus was reported as the best Threshold criterion (TF=2) with an accuracy of about 0.38, whereas the scientific corpus had term frequency Threshold criterion (TF=3) with an accuracy of 0.70. For scientific corpus, the accuracy was 32% higher. The scientific corpus had an average precision improvement of +27% whereas the literacy corpus had +12%. The scientific corpus had an average recall improvement of +26% whereas the literacy corpus had +10%. The F1-measure for scientific corpus was +28% and that of literacy corpus was +11%. The hybrid approach was the more preferable approach for improvements to the classification system, with the average improvement for the two corpora about +16%. The performance of the training set size was evaluated by incorporating separate trial runs on the training set that had a varied number of documents. There were different tested sizes for both the corpora training documents. The paper stated that as the training documents increased, the accuracy also increased.

For the literacy corpus, the global improvement of the performance was about +15%. The stability on the recall measure was noticed from the training document size 191. Similarly for the scientific corpus, the global improvement of the performance was about +26.5%. By comparing the percentage of the evaluation results, the scientific corpus gave the best result.

Yan Li & Chen (2012) tried to develop a text classification system for Chinese documents. The HTF-WDF algorithm was proposed for feature selection. This method was different from other feature selection algorithms which considered the effect of term frequency. The idea of 'fuzzy features', which are the High Term Frequency (HTF) and Weighted Document Frequencies (WDF), were introduced to reduce the problem of the traditional Document Frequency (DF) method. Support Vector Machine (SVM) was used for training the classifier. The documents were represented by using Vector Space Model (VSM). The Chinese documents collected by Fudan University were used as training and testing data sets to verify the accuracy of the proposed algorithm. These documents were classified into twenty categories including Art, Computer, Sports, Environment, Agriculture, Economics, Politics, and Space. The documents in every category were made into two subsets of training and testing. The software, ICTCLAS (Institute of Computing Technology, Chinese Lexical Analysis System) was used to perform word segmentation, part-of-speech tagging on the original data. Every document was represented using VSM. With the application of the HTF algorithm, the terms with high term frequencies were joined. As a result, the documents were converted into vectors which consisted of terms appearing in the document and also the terms with high occurrence frequencies in the document. Then the classifiers of every category were constructed. All the terms in the document did not reflect the topic of the category. The WDF method was used to conduct feature selection for choosing the significant terms in classification to improve the old-style DF algorithm's accuracy. Using WDF method, the terms used in every category were ordered by their weighted document frequencies (WDF). The terms that had $TF' = 0$, and those with $TF' = 1$, were ordered in descending order according to their respective WDF. The first 500 terms with $TF' = 0$ and first 100 terms with $TF' = 1$ were reserved to represent the category. Each category was indicated by the vector that consisted of selected terms. Then finally every document in the training and testing set was converted into numeric data. Software called LIBSVM was used to train the classifier of every category, while the accuracy of the classifier was verified using the testing set. The small numbers of

documents were not sufficient to train the classifier, where out of twenty only eight categories, including 500 documents, were used as testing data. In the experiments, the traditional DF algorithm was compared with the proposed WDF method. Then the performance of the HTF-WDF method was checked. There were different numbers of documents in the testing sub-set. The recalls measured for Arts using DF, WDF and HTF-WDF were 90.43%, 90.70% and 91.37% respectively. The recalls measured for Space using DF, WDF and HTF-WDF were 86.74%, 87.79% and 90.72% respectively. Likewise, for the other six categories recalls were measured using DF, WDF and HTF-WDF. The paper stated that the original DF method upgraded with HTF-WDF yielded better accuracy.

From the above reviews of text classification, the authors Yan Li & Chen (2012),and Harrang, El-Qawasmeh, & Pichappan (2009), dealt with Chinese and Arabic datasets in their experiments respectively. Krishnalal et al. (2010), Al-Mubaid & Umair (2006) and Ting, Ip, & Tsang (July 2011) dealt with Newsgroup datasets. Amongst these authors, Al-Mubaid & Umair (2006) dealt with other datasets WebKB, and Reuters-21578. Hassan, Rafi, & Shaikh (2012) used knowledge repositories namely Wikipedia, Wikitology, Word Net, and the Open Project Directory (OPD) as datasets. Other authors from the reviews used multiple categories from different papers such as physics, chemistry, and AI for text classification.

The challenge or a sub-problem in the classification problem is that a majority of real life problems of classification tasks involve imbalanced data. This problem arises when the classification categories are not uniformly or equally represented. As most of the classification algorithms consider a balanced training set, this issue becomes a partiality towards a majority of the class or categories. Many authors have tried to solve this issue using several machine learning techniques or machine learning algorithms. This is a real-world problem, as a categorization problem is usually to “rule in or rule out”. This usually ends up with the “rule out” category much larger than the former. The next section describes the works that have focussed on classification problems on imbalanced data.

2.3 Reviews of Works on imbalanced datasets

Olson (2005) wanted to examine the relative impact of skewed datasets on common data mining algorithms for two types of data, categorical and continuous. The basic data mining algorithms used in his experiments were decision-tree, regression-based

and neural network models. Three simulated data sets were used based on common applications of data mining in business. The first dataset contained loan applicants; the second contained insurance claims, and the third contained records of job performance. From the experiments, the loan application and insurance fraud datasets had binary outcome while the third had four possible outcomes. The important finding of this paper was that when the data is highly imbalanced, the algorithms tend to degrade by allocating all cases to the common outcome. The author even tried to identify the counter problem when the data is balanced. Oversampling and under sampling were carried out on data in order to balance the data. It was found that the accuracy tended to decline when the data is balanced. The training size reduces if the data is balanced. This leads to the degeneration of the model failure by omitting the cases met in the test case.

The loan data consisted of information about loan applicants. The dataset had 650 past observations of which 400 were used for the full training set and 250 were used for testing. All payments received on time were identified by the variable "1" and by "0" if not received on time. The insurance fraud data consisted of 5000 past claims of which 4000 were for training and 1000 were for testing. The job applicants' data involved 500 past job applicants, of which 250 were used for the full training set and 250 were reserved for testing.

On categorical loan data, the author again used the decision tree, logistic regression and the neural network models. The tests were conducted on continuous data using the same three models. Using loan application categorical data with the decision tree model yielded an accuracy of 0.83 to 0.92 when the training set was from 150 to 400. Loan application continuous data could have yielded better accuracy but it was prone to error when smaller training sets were induced. This was one of the issues considered. The insurance fraud data was highly imbalanced having 60 late cases in the training set of 4000, where late cases designated not paying off insurance on time.

On categorical fraud data, the author used the same set of algorithms as the loan data. The best fit was obtained with the logistic regression and neural network models using categorical data. Exceptions were met with the logistic regression and neural network models over continuous data. Iteration with too small datasets yielded the reverse problem. When the training datasets were balanced for both continuous and categorical data, the error rate was high using the neural network model. The same problem was

identified using the logistic regression model over categorical data. Degeneration of the correct classification rate was noticed when the training set was reduced.

In case of fraud data, the correct proportion of classification was determined and was increased when the training set size increased. The correct proportion was found to be 0.25 when the training size was 28. Smaller training sets had lower correct classification rates. With logistic regression of training sets sized 140 and 250, the classification rates were 0.58 and 0.60 respectively. With decision tree, the classification rates were 0.50 for both training sets of 140 and 250. With smaller training sets the results were inferior and degenerated. The neural network model over categorical data yielded good results and were much more stable for smaller datasets. Results for the regression tree model over continuous data were inferior to those of decision tree over categorical data except for the largest training set. Discriminant analysis over continuous data performed better and did not degenerate even on smaller datasets. Use of the neural network model over continuous data was unreliable but was better with datasets that had more training observations.

The logistic regression had best overall fit using the full training set. This model did not perform well when the dataset was reduced to a point where the training set did not include the cases that appeared in the test set. The author concluded that the Decision tree algorithms were found to be strong according to the degree of balancing applied over datasets. The author encouraged similar tests with real data which would have been much more effective but the characteristics of data mining algorithms or tools were prohibitive.

In this paper Yanling Li, Sun, & Zhu (2010) described imbalance of training data as some classes or categories having more samples compared to other. The standard classification tends to over-adapt to big classes and disregard small classes. For example, in monitoring public opinion, information security or supervision, usually the text numbers holding positive views are more than the text numbers holding negative views. Many applications like medical diagnosis, intrusion detection, and risk management face data imbalance problems. The paper states that many common classifying algorithms such as SVM, *kNN*, and neural network are non-adaptive to dealing with imbalanced data. Therefore, according to this paper, not applying neural networks may yield better results with respect to balancing datasets as discussed by Oslon (2005). Yanling Li et al. (2010) reported that the requirement of having the

correct classification of small categories is higher in practical applications. For example, in the application of bad information filtering, the number of bad information is less than the number of general information. Bad information is described as the information that is not being exposed on the internet. The internet hiding information from us arises due to the use of personalization engines by Google or Netflix to filter the information. However the aim of information filtering remains to identify the bad information and sieve them. This led to urgent attention being paid to the data imbalance problem in the field of text mining. Aiming to solve the ever present problem of imbalanced data in text classification, Yanling Li et al. (2010) studied several forms of imbalanced data such as text number, class size, subclass and class fold. The authors said that the difference in the number of samples between big and small categories may vary in the ratio of 1:100, 1:1000 or even 1:10000. This difference in the number of samples can lead to lower classification performance. Weiss (2004) and Joshi (2002) report that performance of the classification is lowered not only due to skewed data but also sample size of small categories, existence of subclass in the category or even independence of samples. Other than difference in the text number of various classes, elements such as class size, sub-categories, and category overlap can lower classification performances. Class size is the text number in one class. Class overlap means there is overlap between the categories or the class. A sub-category is where many subclasses exist in one class and if the numbers of subclasses are not the same as the class, then class the problem of class imbalance arises.

Yanling Li et al. (2010) carried out a series of experiments using the experimental data collected from posts downloaded from People's Network powerful forum. Based on the views of the post content, posts were divided into two categories as C1 and C2. The category C1 represents high house prices. The category C2 is where house prices are not high. The performance of the classification based on the experiment was measured by using three common assessment indexes: Precision (P), Recall rate (R) and F1 test value. For the overall classification result, the macro average mode (classification accuracy using permutations) was accepted. The mean value was calculated on the classification results of all the categories to achieve macro-p, macro-r and macro-F1. This paper mainly aims to analyse various forms of the data imbalance problem in text classification, such as text distribution, class overlap, and class size, through a succession of experiments. This research can be helpful in investigating the cause of poor classification performance using imbalanced data.

As already mentioned, an imbalanced dataset is often a problem in real world applications as it causes poor classification performance when using machine learning algorithms. Many attempts have been made to deal with the classification of imbalanced datasets. In Ganganwar (April 2012), the author presents a brief review of existing solutions to the class imbalance problem attempted both at the data and algorithmic levels. A number of solutions were proposed previously by the researchers Chawla, Japkowicz, & Kolcz (June 2004) both at data as well as algorithmic levels. The paper states that imbalanced data can be rebalanced using oversampling or undersampling techniques. Oversampling duplicates examples in the minority or smaller class while undersampling eliminates examples in the majority or larger class. This paper asserts that a dataset is called imbalanced if it has many more samples from one class than from the rest of the classes. Datasets are imbalanced when at least one class has only a small number of training examples (minority class) and the other classes constitute a majority. The paper states that classifiers have good accuracy on the majority class but poor accuracy on the minority class which is due to the influence that the larger majority class has on traditional training criteria. Most researchers such as Vajda & Fink (2010) have agreed that unequal class distribution or imbalanced data has led the performance of the classifiers to be unfair towards the majority class. Krishna Veni & Sobha Rani (2011) pointed out that the performance of the existing classification algorithms are poor because they are accuracy driven, i.e., they tend to minimise the overall error in which the minority class contributes very little. The algorithms assume equal distribution of data for all the classes and also assume the errors coming from various classes have the same cost. The paper states that at data level sampling is the common approach to deal with imbalanced data. Hybrid algorithms based on decision tree (DT), particle swarm optimisation (PSO) and feature selection proved to be better performers than just applying sampling techniques as oversampling or under sampling. This paper also states that cost sensitive classifiers can be proposed to deal with imbalanced data at the algorithmic level. The algorithms can be modified SVMs, neural networks genetic programming algorithms, probabilistic decision tree and learning methods.

In the past studies efforts were made to solve two-class imbalance problems. Wang & Yao (August 2012) posed a challenge to solve the multiclass imbalance problem that exists in real-world applications. The paper reports that class imbalance problems have drawn recent attention as classification difficulty is generated due to imbalanced class

distributions. Many ensemble methods have been utilized to solve such imbalances. Many efforts were made to solve two-class imbalance problems. This paper proposes that the generalisation ability of some ensemble algorithms can be the solution to these problems as well as newly proposed algorithm named as Adaboost.NC. The Adaboost.NC can handle the multiclass imbalance problem directly and effectively. This paper studied the impact of multi-majority and multi-minority on the performance of two basic resampling techniques. Multi-majority and Multi-minority have both strong negative effects but Multi-majority is more harmful on the generalization performance. As Adaboost.NC yielded a better performance in many real-world multiclass imbalance tasks, this algorithm was compared with other ensemble methods. Adaboost.NC was better while balancing the classification performance among majority and minority classes through G-mean instead of using class decomposition.

The studies discussed above mainly highlight an overview of the classification algorithms, related works on text classification and impact of imbalanced datasets in text classification. The classification algorithms mainly relate to the Decision Trees model, Maximum Entropy model, Perceptron and Nearest Neighbour classification rule. Furthermore, the functions of the algorithms are discussed and how they work on the data. The related works on text classification mainly described the different machine learning techniques or machine learning algorithms, namely Naïve Bayes (NB) with SOM, *kNN*, Adaboost, HMM(Hidden Markov Model), Decision Trees, Matrix Regression, and SVM using SMO function in WEKA. These different algorithms impacted on various datasets used which were mostly imbalanced. The datasets were: from various resources such as 20News Group, RCV1, and Reuters-21578; Information from knowledge repositories such as Wikitology, the Open Project Dictionary (OPD), and the Wikipedia database; and from paper abstracts like Physics (PH), Chemistry (CH), Artificial intelligence (AI), Educational Engineering (EDE). On the basis of the research goals of the related articles, the authors performed different sets of experiments. Some of the articles compared the classification accuracies in terms of different datasets using different algorithms. Some of the articles compared the classification accuracies in terms of categories in the dataset using different algorithms. Most of the articles which reviewed the impact of imbalanced datasets tried to identify the classification accuracies using different algorithms on imbalanced datasets. They also managed to balance the datasets to achieve better accuracies, addressed the issues

arising due to data imbalance, tried to identify the cause of imbalance and suggested some ideas for further development.

This research thesis also carried out some experiments using four upper level topic codes from RCV1-v1 as input dataset. This paper tried to identify the effects of imbalanced data on document classification algorithms. One balanced dataset and two imbalanced datasets were used in the experiments. First the balanced dataset was considered as the benchmark to see if the classification accuracies make any difference using the implemented algorithms in MALLET - Naïve Bayes, Balanced Winnow, Max Ent, Max Ent L1 and MC Max Ent. Then the impact of imbalanced datasets was identified using the stated classification algorithms. The classification accuracies were compared with the two imbalanced datasets and the best algorithm was chosen.

Chapter 3. *Methodology*

This chapter provides the methodology used during research for this thesis. This methodology includes details on the tools used, input data used including pre-processing of the data and formation of datasets, algorithms used, and experimental setup. Section 3.1 describes the tool MALLET used for the experimental performance of this thesis. Sub-section 3.1.1 describes the installation procedure of MALLET. Section 3.2 outlines the input data-the Reuters corpus used for performing the experiments of this thesis. Sub-sections 3.2.1 describe the pre-processing of the Reuters data as MALLET input data and that of 3.2.2 describe about the datasets formation for carrying out experiments. Section 3.3 provides a brief idea about the algorithms used in the experiments. Sub-sections 3.3.1 to 3.3.3 provide a more detailed description of each of the algorithms used in the experiments. Section 3.4 outlines the experimental set-up and sub-section 3.4.1 describes the experimental procedure.

3.1 Tools Used

This section describes the tools used for the experiments. MALLET (MAchine Learning for LanguagE Toolkit) is a topic modelling package developed by Andrew Kachites McCallum, as the primary author of the package, in 2002. Many graduate students and staff from the University of Pennsylvania contributed to the development of this package.

MALLET is open source software used for statistical natural language processing, clustering, document classification, information extraction and topic modelling. It has java API for several information processing tasks and can be easily integrated into higher level text applications.

MALLET also contains sophisticated pre-processing tools for document classification which includes routines for converting text to features or numerical presentations for efficient processing. This process is implemented through a flexible system of “pipes” that handles different jobs like string tokenization, stop-word removal and conversion of sequences to count vectors. These pre-processing tools convert raw data into a format

required for MALLET. Data in MALLET is presented as list of instances and each instance contains a data object. An instance may be a name, or in classification context, a label. For importing data in MALLET format, two formats can be used. In the first, the data is contained in separate files in directories corresponding to the categories. In the second format the data is contained in a single file, containing one instance per file including the categories. The core of the classification tools includes various algorithms like Naïve Bayes, Decision Trees, and Maximum Entropy among several others. MALLET also includes codes to measure classifier performance using numerous frequently used metrics such as Accuracy, Precision, Recall and F-Measure. Also included in MALLET are tools for sequence tagging which comprises of algorithms such as Hidden Markov Models, Maximum Entropy Markov Models, and Conditional Random Fields. The topic modelling toolkit in MALLET contains sample-based implementations of Pachinko Allocation, Latent Dirichlet Allocation (LDA), and Hierarchical LDA, through which unlabelled text can be analysed. Most of the algorithms in MALLET perform numerical optimization which finds parameters to maximize a log-likelihood function on observed data. In addition, MALLET also has an add-on package called GRMM that provides a graphical interface to train CRFs with arbitrary structure (McCallum, 2013).

3.1.1 Installation of MALLET (mallet-2.0.7)

For Windows installation, mallet-2.0.7 can be very easily installed by simply unzipping the packed file and setting an environment variable MALLET_HOME to point to the unzipped directory. Once the MALLET was ready to use, a working folder named as "mdata" was made manually under mallet-2.0.7 to build up different working subfolders for different datasets.

3.2 Input Data Used

This section describes the input data used in the experiments for the research of this thesis. The data was taken from Reuters Corpus data (RCV1). The following brief description of Reuters Corpus, is described briefly below are taken from Lewis, Yang, Rose, & Li (2004).

The Reuters Corpus data (RCV1) containing news articles was provided by the research supervisor. The data is available on two CD-ROMs as volume I and volume II and formatted in XML. RCV1 can be easily obtained from online databases which are a modified version of the Reuters-21578 as it contains 35 times as many documents and

60 times more reliable coding compared to Reuters-21578 according to Lewis et al. (2004). The coding is used to categorise the documents. RCV1 is an archive of more than 800,000 manually categorised newswire stories. These stories are made available by Reuters Ltd. for research purposes. Reuters is considered to be one of the world's largest text and television agencies and produces an about 11,000 stories in 23 languages in a day, according to Lewis et al. (2004).

The Reuters Corpus Volume I and Volume II are generally referred to as RCV1-v1 and RCV1-v2 respectively. RCV1-v1 contains the original data whereas RCV1-v2 contains the edited articles. RCV1-v1 contains 806,791 documents and RCV1-v2 contains 804,414 documents. Stories are organised into one article per file with unique document IDs. IDs start from 2286 to 810597 for RCV1-v1 and 2286 to 810596 for RCV1-v2 respectively. There are some gaps in the ID numbers because of some deleted files.

The RCV1 corpus uses XML formatting to encode metadata. The archiving process and preparation of the XML dataset includes considerable validation and verification of the content. The data also includes regularization of heading, by-line formats, copyright statements, and attempts to remove duplicated documents.

To retrieve data from RBB (Reuters Business Briefing), category codes were allocated to stories from three topic sets. RBB is a world online business news and information database that contains information about current news and historical news for companies. On the RCV1 CD the file "*topic_codes.txt*" has 126 codes. Out of these, 11 codes labelled as 'current news' and 2 codes named as 'temporary' are all idle in the corpus data (Rose, Stevenson, & Whitehead, 2002). As an example, the list for topic codes is provided in Appendix A.

Code sets were designed to categorically organise the data in the following ranked code groups:

- CCAT represents Corporate or Industrial data
- ECAT represents data on Economics
- GCAT represents Government or Social data
- MCAT represents Markets data

3.2.1 Pre-processing of Reuters data as MALLET input data

For the purposes of this research four upper level topic codes were chosen from RCV1-v2 for the classification experiments. They are CCAT, ECAT, MCAT and GCAT. The articles or the documents in XML formats could belong to more than one topic codes, for example, to ECAT and MCAT. An example of the XML document belonging to more than one category is provided in Appendix B. However, the articles that belong exclusively to only one category were chosen and separated into folders CCAT, GCAT, MCAT and ECAT using an XML Parser tool provided by the supervisor. The XML Parser tool examined all the documents and randomly chose the articles that belong to only one category. This is to help facilitate algorithms used in the experiments to perform binary classification task rather than classification into multiple categories. The XML data in the original format was converted into pure text files and separated into folders corresponding to their respective categories. An example of a data file is provided in Appendix C. The conversion of the XML files into text files is shown schematically in Figure 1. The text files are written in English language.

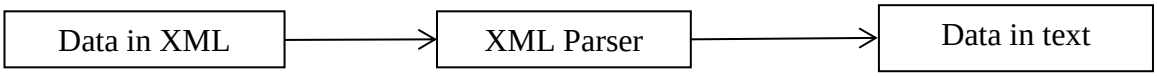

Figure 1. Conversion of data from XML to text format

The number of articles belonging to each of the categories is given below:

- CCAT contained about 173878 files
- ECAT contained about 31848 files
- GCAT contained about 99539 files
- MCAT contained about 87857 files

The data for the binary classification task was chosen from the CCAT category while the second set was chosen from a random mixture of files from the rest of the 3 categories, ECAT, GCAT and MCAT.

3.2.2 Formation of datasets

Generally, the datasets can be balanced or imbalanced. In the real world when huge datasets are considered, they may be highly imbalanced or can occur in unknown proportions. The imbalance problem depends upon the amount of data and occurs when some specific problem is to be solved. For example the imbalance problem is substantial in the case of human genome data.

For this thesis, three working subfolders for balanced and imbalanced datasets were created under the working folder “mdata” discussed in section 3.1.1. One working subfolder was for balanced datasets and the other two were for imbalanced datasets. This is because balanced datasets had two categories having equal proportion of data whereas each imbalanced dataset had the same two categories with unequal proportions of data in each. The effects on testing accuracies using various common classifying algorithms were observed on increasing the amount of training data for both balanced and imbalanced datasets. The balanced dataset had similar alphabet size and features (vocabulary) in each category CCAT and OTHERS. The two imbalanced datasets had substantially different alphabet sizes.

Most machine learning algorithms do not support concurrent classification into multiple categories. However this is not a limitation as such tasks can be decomposed into binary classification tasks and iterated over the rest of the categories.

The binary classification was done on two categories, CCAT and OTHERS, which consisted of a random mixture of files from the ECAT, GCAT and MCAT categories. Initial exploration on experiments were done on balanced or equal data in each of the categories, with the objective of finding the point of over-training and also as a benchmark for comparison with imbalanced or unequal datasets. The experiment on the imbalanced dataset was performed firstly by increasing the number of files in the OTHERS category and keeping the number of files in the CCAT category constant. A second experiment was also carried out by increasing the number of files in the CCAT and keeping the files in the OTHERS constant. This was done to ascertain if the features (vocabulary) in each of the categories had any impact on the classification performance.

3.3 Algorithms Used

This section describes the algorithms that were chosen to be used for the experiments. The algorithms used for the experiments were Naïve Bayes, Balanced Winnow, and three variations of Max Ent, namely Max Ent, Max Ent L1 and MC Max Ent supported by MALLET. Mallet also contains some other algorithm implementations such as AdaBoost, Bagging and C45 classifier. However, these algorithms either gave similar results or resulted in errors due to unresolved bugs, or in the case of the C45 algorithm, the experiments took too long as it is a computationally expensive algorithm. A brief

description of each of the algorithms used is given below including its application for text classification.

3.3.1 Naïve Bayes Classifier

The **Naïve Bayes** classifier is a probabilistic classifier based on Bayes theorem. The parameters of the generative model of Bayesian Classifiers can be estimated by using supervised learning on training examples. Classification on new examples can be performed using Bayes' rule. This is done by selecting the category that is most expected to have created the example. The Naïve Bayes assumption assumes that all the topographies of the examples are sovereign of each other, given the context of the category. Although this assumption is not entirely true in the real world, nevertheless Naïve Bayes generally performs very well in typical classification tasks. In practical applications, Naïve Bayes has good accuracy with a variety of datasets. The Naïve Bayes classifier is considered to be more efficient than other supervised learning classifiers because of the independence assumption, which allows constraints for each feature to be learned discretely. The number of features in document classification is commonly proportional to the training document sets vocabulary size (Ramdass & Seshasai, 2009). The application of the Naïve Bayes classifier can be examined in McCallum & Nigam (1998), where two different first-order probabilistic models were used and compared for classification. The models used were the Bernoulli model and the Multinomial model. The Bayesian Network using the Bernoulli model does not have dependencies between words and binary word features. The Multinomial model, alternatively the uni-gram language model, deals with integer word counts. The paper mainly compares the theory and practice of two different first-order probabilistic classifiers which make Naïve Bayes assumptions. McCallum & Nigam (1998) found that the Multinomial model was uniformly better than the Multi-variate Bernoulli model. Five types of real-world corpora were used where the Multinomial model reduced error by an average of 27%, and occasionally by more than 50%.

Probabilistic Model for Naïve Bayes

The probability model for the Naïve Bayes classifier is denoted by p (C|F1, ... , Fn) over a dependent class variable C which shows one of the possible classification labels that can be conditioned on feature variables from F1 to Fn. According to the Naïve Bayes Theorem this can be rewritten as:

$$p(C|F1,\ldots,Fn) = \frac{p(C)p(F1|C)p(F2|C,F1)\ldots p(Fn|C,F1,F2,\ldots,Fn-1)}{p(F1,\ldots,Fn)} \quad (1)$$

By using the Naïve Bayes assumption, the conditional probabilities for each feature F_i in the numerator for the equation above become conditional only on C.

Now the above equation can be rewritten as:

$$p\,(C|F1,\ldots,Fn) = \frac{1}{Z\,p(C)\pi_{i=1}^{n}\,p(Fi|C)} \quad (2)$$

Z is considered as the normalization factor to guarantee valid probability distribution (Ramdass & Seshasai, 2009). The above probabilistic model for Naïve Bayes can be illustrated with the help of an example. Thornton (n.d.) considered sample ML (Machine Learning) tasks such as:

SYMPTOM	OCCUPATION	AILMENT
Sneezing	nurse	flu
sneezing	farmer	Hay fever
headache	builder	concussion
headache	builder	flu
sneezing	teacher	flu
headache	teacher	concussion
sneezing	builder	???

(Thornton, n.d.)

The question that arises is "what ailment can be predicted for a sneezing builder and why?" To achieve an answer to this question, it is better to use a probabilistic method such as the Naïve Bayes Classifier.

Probabilities

For example, consider the data that lists symptoms and ailments for everybody in a definite group.

SYMPTOM	AILMENT
sneezing	flu
sneezing	Hay fever
headache	concussion
sneezing	flu
coughing	flu
backache	none
vomiting	concussion
crying	Hay fever
temperature	flu
drowsiness	concussion

(Thornton, n.d.)

Prediction:

There are 10 cases in all. So the probability can be worked out by selecting a particular ailment or symptom and just by counting and dividing by 10.

P (Hay fever) = 2/10 = 0.2

P (vomiting) = 1/10 = 0.1

Using simple, statistical models of the data, these (so-called prior) probabilities can be used for prediction.

At this point, it remains undecided that whether someone has flu or Hay fever.

The fact that P (flu) > P (Hay fever) can be used to predict that the ailment is more likely to be flu.

Conditional probabilities

This type of modelling becomes more useful when conditional probabilities are used.

These are the values worked out by determining the probability of observing one value given another is observed, e.g., the probability of vomiting given concussion.

Conditional probabilities are denoted using the bar `|' to separate the conditioned from the conditioning value.

The probability of vomiting given concussion is written as

P (vomiting|concussion)

This value can be worked out by identifying what proportion of the cases involving concussion also displays vomiting.

P (vomiting|concussion) = 1/3 = 0.3333

Prediction from Conditional Probabilities

Conditional probabilities facilitate conditional predictions.

For instance, someone who is known to have concussion can be told that there is a 1/3 chance of them vomiting, which can be one of the ways of producing diagnoses.

If someone has been sneezing a lot, then it can be said that there's a 2/3 chance of them having flu, as P (flu|sneezing) = 2/3

There is slightly less likelihood (1/3) that they have hay fever, as P (Hay fever|sneezing) = 1/3

The Problem of Multiple Attributes

The question which then arises is what would happen if the data includes more than one attribute, which can be shown as follows:

SYMPTOM	OCCUPATION	AILMENT
sneezing	nurse	flu
sneezing	farmer	hay fever
headache	builder	concussion

(Thornton, n.d.)

The probabilities conditional can be worked out on multiple symptoms such as P (flu|sneezing,builder).

If the combination does not appear in the data, then the question arises as to how the conditional probability can be calculated.

Using Inversion

The probability conditional cannot be sampled on a combination that does not appear, but the probabilities that do appear can be worked out.

Observable probabilities that contribute to P (flu|sneezing,builder) are:

P (flu)
P (sneezing|flu)

P (builder|flu)

These attributes can be put together in some way.

The Naïve Assumption

The Probability theory says that if various factors do not depend on each other in any way, then the probability of seeing them together can be just the product of their probabilities.

So it can be assumed that sneezing does not have an impact on whether a person is a builder. Then it can be said that:-

P (sneezing,builder|flu) = P (sneezing|flu) P (builder|flu)

The probability of a sneezing builder having flu must depend on the chances of this combination of attributes indicating flu. As a result, P (flu|sneezing,builder) must be proportional to P (flu) P (sneezing,builder|flu).

Normalization Needed

The value is purely based on the cases of flu. It does not take into account how common the ailment is. Therefore, the probability of this combination of attributes relating to flu in particular should be factorised, rather than some other ailment. The probability can be factorised by expressing the value in the proportion to the probability of seeing the grouping of attributes.

$$P(\text{flu} \mid \text{sneezing,builder}) = \frac{P(\text{flu})P(\text{sneezing,builder} \mid \text{flu})}{P(\text{sneezing,builder})}$$ (Thornton, n.d.)

Thus, the value wanted can be achieved.

The constituents which are needed can be assembled.

P(flu)=0.5
P(sneezing|flu)=0.66
P(builder|flu)=0.33
P(sneezing,builder|flu)=(0.66x0.33)=0.22
P(sneezing)=0.5
P(builder)=0.33
P(sneezing,builder)=(0.5x0.33)=0.165

The values are functioned in to the formula:

$$\frac{0.5 \times 0.22}{0.165} = 0.66$$ (Thornton, n.d.)

It indicates that the sneezing builder has flu with probability 0.66 (Thornton, n.d.).

3.3.2 Balanced Winnow

The **Winnow** algorithm is a member of online-mistake driven family of algorithms like the Perceptron. This algorithm does not identify the linear separation among examples allocated to different categories "additively" but somewhat "multiplicatively". The categorisation system of patent documents (i.e. documents having descriptions of invention and usefulness) is based on a variation of the Winnow algorithm called **Balanced Winnow** which is described in Koster, Seutter, & Beney (2003). Since the standard implementation of Balanced Winnow only work on binary class problems, ensemble learning can be applied iteratively by transferring the multi-category problem into ensemble of binary problems. The Balanced Winnow classifier consists of positive and negative weight pairs. These weights are used to calculate the class membership score of a document. The positive weights provide proof for class membership while the negative weights provide negative proof. The difference between positive and negative weights indicates the overall weight of a feature. This weight of a feature can be updated only when a mistake occurs according to Beuls et al.(2010).

If a mistake occurs on a positive example, the positive part of the weight is endorsed while the negative part of the weight is lowered. If a mistake occurs on as a negative example, then the positive part of the weight is lowered while the negative part of the weight is endorsed. Other than the promotion and demotion such as α and β respectively, this algorithm also has a threshold θ that decides the class membership. In addition a thick threshold is also used. This means that in training rather than ranking the score documents above (1) or irrelevant documents below (1), i.e. θ, there can be two possible thresholds such as $\theta^+ > 1.0$ and $\theta^- < 1.0$. The result can be implied as incorrect if the score of a document is below θ^+ and it belongs to the class or if the document does not belong to the class although it's score is above θ^- as indicated by (Beuls et al., 2010).

3.3.3 Max Ent

Maximum Entropy modelling is referred to as Max Ent. In general, there can be two classification problems namely single-labelled and multi-labelled classification problems. In the case of single-labelled data classification, each data point is reciprocally exclusive and belongs to one category, whereas in multi-labelled data classification, each data point is not reciprocally exclusive and belongs to more than one category. At the present time, multi-label classification problems are decomposed into numerous binary classification problems and the final labels are determined by combining the classification results from all the binary classifiers. To propose a multi-labelled data classification method, the mutual correlations between data categories can be modelled using the maximum entropy model. A conditional probability model such as $\Pr(y \mid x)$ is constructed from the training dataset where x is considered as the feature vector for input data point and y is considered as the class membership vector. Here, y has each element y_i which shows whether x belongs to i'th class or not. The Max Ent (ME) method estimates the parameters during the model construction process. Regularised parameters can be introduced to show the errors between empirical and real distributions and overfitting problems can be avoided as well (Zhu, Ji, Xu, & Gong, 2005). The application of the Maximum Entropy classifier can be examined in Nigam, Lafferty, & McCallum (1999). Three types of Max Ent classifiers have been used in this thesis, namely Max Ent, MC Max Ent and Max Ent L1. MCMax Ent stands for Maximum Entropy Classifier. MaxEnt L1 is a Multivariate logistic Regression Classifier having regularization Method L1. L1 regularization disciplines the weight vector for its L1- type. L1- type is the sum of the absolute values of the weights.

3.4 Experimental Setup

This section describes the experimental procedure.

3.4.1 Experiments

This section deals with experiments carried out on balanced and imbalanced datasets. The training, validation and test data splits were done randomly according to the proportions specified for the experiments. Details of these splits are shown in Appendix D.

Experiment (A) - Balanced datasets

This data set consisted of test files (Te) which were kept arbitrarily constant at 1000 with 500 files in each of the CCAT and OTHERS categories. The validation files (Va)

were also kept arbitrarily constant at 400 having 200 files in each of the CCAT and OTHERS categories. The training files (Tr) were incremented equally and arbitrarily from 20 through 6100 for both the CCAT and OTHERS categories.

Table 1 shows the arrangement of files in categories CCAT and OTHERS for Balanced datasets.

Table 1. Arrangement of files for balanced datasets

Inc.No.	Test files (Te)		Validation files (Va)		Training files (Tr)		Total files
	CCAT	OTHERS	CCAT	OTHERS	CCAT	OTHERS	
1	500	500	200	200	10	10	1420
2	500	500	200	200	30	30	1460
3	500	500	200	200	50	50	1500
4	500	500	200	200	70	70	1540
5	500	500	200	200	90	90	1580
6	500	500	200	200	110	110	1620
7	500	500	200	200	130	130	1660
8	500	500	200	200	1000	1000	3400
9	500	500	200	200	2000	2000	5400
10	500	500	200	200	3000	3000	7400
11	500	500	200	200	3010	3010	7420
12	500	500	200	200	3030	3030	7460
13	500	500	200	200	3050	3050	7500

Experiment (B) - Imbalanced datasets

As for the previous experiment the data set consisted of test files (Te) which were kept arbitrarily constant at 200 with 100 files in each of the CCAT and OTHERS categories. The validation files (Va) were kept constant at 100, with 50 files in each of the CCAT and OTHERS categories. The training files (Tr) of CCAT were kept constant arbitrarily at 50 and that of OTHERS were incremented from 20 through 5000.

Table 2 shows the arrangement of files in categories CCAT and OTHERS for Imbalanced datasets.

Table 2. Arrangement of files for imbalanced datasets

Inc. No.	Test files (Te)		Validation files (Va)		Training files (Tr)		Total files
	CCAT	OTHERS	CCAT	OTHERS	CCAT	OTHERS	
1	100	100	50	50	50	20	370
2	100	100	50	50	50	30	380
3	100	100	50	50	50	50	400
4	100	100	50	50	50	100	450
5	100	100	50	50	50	200	550
6	100	100	50	50	50	500	850
7	100	100	50	50	50	1000	1350
8	100	100	50	50	50	3000	3350
9	100	100	50	50	50	5000	5350

Experiment (C) - Imbalanced datasets

As for Experiment (B) on Imbalanced datasets, the arrangement of files in the datasets were very similar except that the training files in the OTHERS category were kept constant arbitrarily at 50 and those in the CCAT category were incremented from 20 through 5000.

Table 3 shows the arrangement of files in categories CCAT and OTHERS for Imbalanced datasets.

Table 3. Arrangement of files for imbalanced datasets

Inc. No.	Test files (Te)		Validation files (Va)		Training files (Tr)		Total files
	CCAT	OTHERS	CCAT	OTHERS	CCAT	OTHERS	
1	100	100	50	50	20	50	370
2	100	100	50	50	30	50	380
3	100	100	50	50	50	50	400
4	100	100	50	50	100	50	450
5	100	100	50	50	200	50	550
6	100	100	50	50	500	50	850
7	100	100	50	50	1000	50	1350
8	100	100	50	50	3000	50	3350
9	100	100	50	50	5000	50	5350

Chapter 4. *Experimental Results*

This chapter reports the results of the experiments described in Chapter 3. The details of the results on balanced and imbalanced datasets are described, including alphabet size, number of files used, testing accuracies over increased training files and classification performance measures.

4.1 Results on Balanced Datasets and Imbalanced Datasets

4.1.1 Alphabet size

Table 4 outlines a sample alphabet size that resulted from Experiment (A). The rest of the alphabet sizes are considered in a similar way for Experiments (B) and (C) for imbalanced datasets provided in Tables (A) and (B) in Appendix E.

The alphabet size is critical because if it is not maintained, then there would be a very large amount of words in a typical training dataset. An excessive alphabet size can lead to very large differences between texts within a category. This underlying difficulty in text categorization can be overcome by using various techniques such as stop words removal, stemming and feature selection (Vert, July 2001).

Table 4. Training, testing and validation alphabet size

Input Balanced data (Equal increment of training files for CCAT + OTHERS)		Training/Testing/Validation alphabet size
CCAT	OTHERS	
10	10	17518
30	30	17733
50	50	18008
70	70	18220
90	90	18403
110	110	18553
130	130	18644
1000	1000	27177
2000	2000	33337
3000	3000	38934
3010	3010	38983
3030	3030	39109
3050	3050	39281

Figure 2. Alphabet size over increasing training files for balanced dataset

Figure 2 shows the rate of increase of alphabet size for increasing numbers of training files for balanced datasets. With the increment of training files for both the CCAT and OTHERS categories, the rate of increase of the alphabet size can be analysed which is indicated in Table 5 .

Table 5. Rate of increase of alphabet size with increasing training files

Training Files	Rate of Increase
20 - 260	4.69
260 - 2000	4.9
2000 - 4000	3.08
4000 - 6000	2.8
6000 - 6100	3.47

From this research, it is evident that the greatest increase in alphabet size is between files 260 and 2000. The rate of increase between 2000 and 6000 is much less than the increase between files 20 and 260, as well as between files 6000 and 6100. It would be also interesting to focus upon this behaviour of increase of files and can be looked vividly in future research.

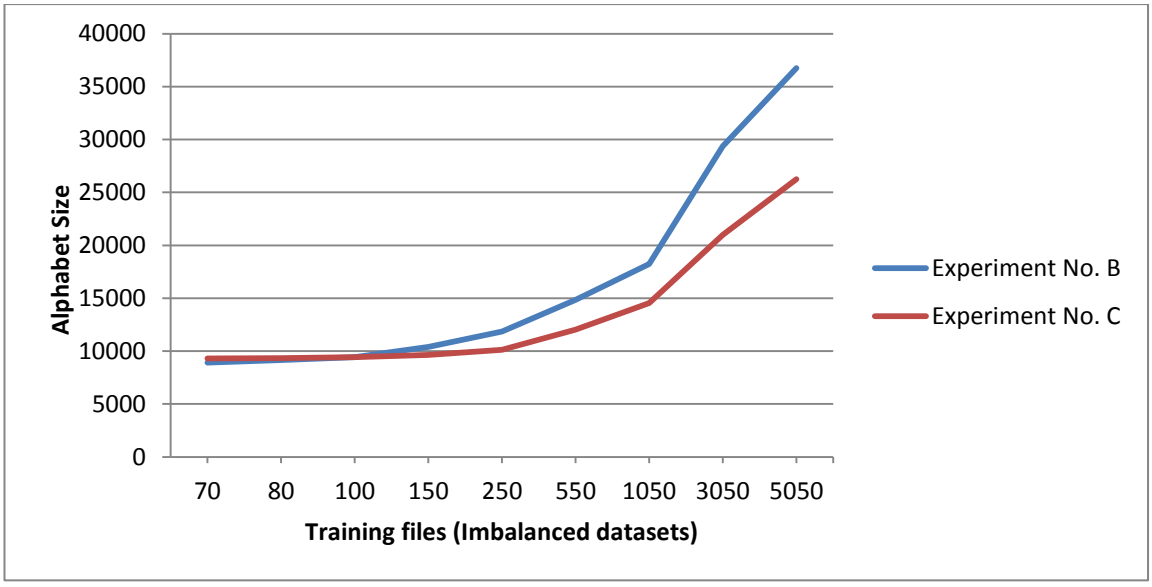

Figure 3. Alphabet sizes between imbalanced datasets

Figure 3 shows the rate of increase of alphabet size for increasing numbers of training files for imbalanced datasets. An obvious observation is that the alphabet size increases with the increasing number of training files, since new words get introduced. The rate of increase and the maximum number of alphabets in the increasing training files in the OTHERS category (blue line) is higher compared to the CCAT category (red line). This is because the OTHERS category contains articles from a wide range of subject areas likely to contain a larger vocabulary compared to CCAT, which contains articles from only one category.

The increment intervals in the number of training files for imbalanced datasets for Experiments (B) and (C) were considered similar, while the increment intervals in the number of training files for balanced dataset for Experiment (A) were considered different. Therefore, a comparison is made with Experiments (B) and (C).

4.1.2 Files used

In this section, the information about the number of training, testing and validation files for balanced as well as imbalanced datasets are provided. The number of files are based on Experiments (A), (B) and (C) shown in Table 6, Table 7 below and Table (C) in Appendix E respectively.

To be able to judge the effectiveness of any classifier, it needs to be tested on data that has not been used for its training. This is referred to as test dataset. Commonly, a classifier is trained with a training set and tested with a test set to investigate the classification accuracy. To improve the performance of the classifier, validation sets or development sets are generally used over training data. The test sets are used because the efficiency of the algorithm is determined not by its performance on the training data but by its ability to perform well on unknown data. The datasets for the experiments were randomly split into training, testing and validation sets. For example, 60% of the data was set for training; 10% was set for validation and 30% was set for testing.

Table 6. Training files, testing files and validation files

Input Balanced data (Equal increment of training files for CCAT + OTHERS)		Total Training Files	Total Validation Files	Total Testing Files
CCAT	OTHERS			
10	10	852	142	426
30	30	876	146	438
50	50	900	150	450
70	70	924	154	462
90	90	948	158	474
110	110	972	162	486
130	130	996	166	498
1000	1000	2040	340	1020
2000	2000	3240	540	1620
3000	3000	4440	740	2220
3010	3010	4452	742	2226
3030	3030	4476	746	2238
3050	3050	4500	750	2250

Table 6 provides information about the equal increment of training files for categories CCAT and OTHERS based on Experiment (A). The table also displays different numbers of training, testing and validation files. These different numbers of training, testing and validation files are due to the splitting of files in definite proportions that were set for the experiments.

Table 7. Training files, testing files and validation files

Input Imbalanced data (Unequal Increment of Training Files for CCAT and OTHERS)		Total Training Files	Total Validation Files	Total Testing Files
CCAT	OTHERS			
50	20	222	37	111
50	30	228	38	114
50	50	240	40	120
50	100	270	45	135
50	200	330	55	165
50	500	510	85	255
50	1000	810	135	405
50	3000	2010	335	1005
50	5000	3210	535	1605

Table 7 provides information about the unequal increment of training files for categories CCAT and OTHERS based on Experiment (B) for imbalanced dataset. Table (C) provides similar information based on Experiment (C) provided in Appendix E.

4.1.3 Testing Accuracies

This section provides information about testing accuracies over increasing training files for balanced datasets (Experiment A). The accuracies were obtained after the implementation of the described algorithms Naïve Bayes, Balanced Winnow, MC Max Ent, Max Ent and Max Ent L1 using MALLET. The testing accuracies over increasing training files are provided in a similar way for imbalanced datasets labelled as Experiments (B) and (C).

Table 8. Testing accuracies over increasing training files

	Testing Accuracies based on selected Algorithms				
Increment of training files	NaiveBayes	Balanced Winnow	MCMaxEnt	MaxEnt	MaxEnL1
20	0.950	0.913	0.917	0.899	0.931
60	0.933	0.936	0.929	0.899	0.929
100	0.955	0.940	0.931	0.931	0.953
140	0.948	0.943	0.939	0.928	0.948
180	0.951	0.943	0.921	0.919	0.962
220	0.936	0.911	0.927	0.899	0.946
260	0.939	0.935	0.927	0.907	0.937
2000	0.950	0.944	0.945	0.934	0.950
4000	0.958	0.949	0.954	0.936	0.951
6000	0.947	0.955	0.943	0.950	0.955
6020	0.951	0.949	0.955	0.956	0.961
6060	0.957	0.958	0.956	0.952	0.967
6100	0.955	0.952	0.956	0.958	0.964

Table 8 shows the testing accuracies for the different algorithms with increasing numbers of training files from 20 through 6100.

The highest and lowest testing accuracies are shown over training files based on individual algorithms in the following Table 9. This information is taken from Table 8.

Table 9. Highest and lowest testing accuracies over increasing training files

Algorithms	Highest testing accuracy over increasing training files (no. of training files)		Lowest testing accuracy over increasing training files (no. of training files)	
Naïve Bayes	0.958	(4000)	0.933	(60)
Balanced Winnow	0.958	(6060)	0.911	(220)
MC MaxEnt	0.956	(6060 and 6100)	0.917	(20)
MaxEnt	0.958	(6100)	0.899	(20, 60 and 220)
MaxEnt L1	0.967	(6060)	0.929	(60)

Table 9 highlights that the highest accuracy of 0.967 was obtained over training file 6060 by Max Ent L1 algorithm. The lowest accuracy of 0.89 was obtained for training files 20, 60 and 220 by Max Ent algorithm.

The data in Table 8 plotted in Figure 4 indicates the effect of the testing accuracies over increasing training files for balanced datasets.

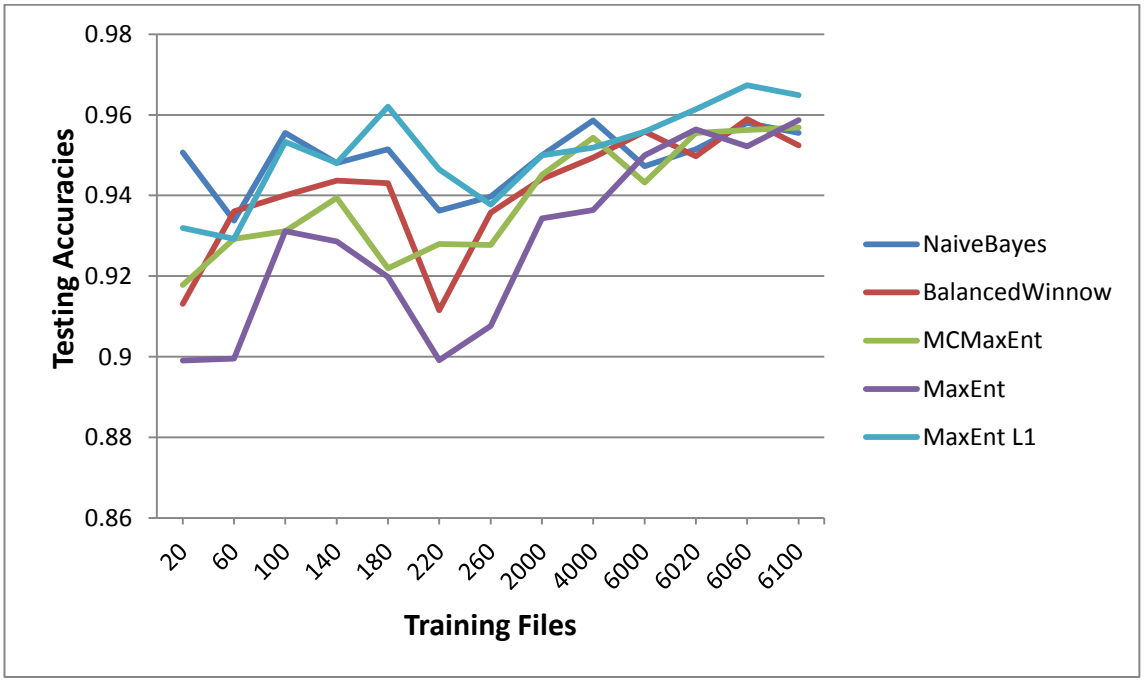

Figure 4. Plot of testing accuracies versus the number of training files for balanced dataset

Figure 4 shows the plots for the accuracies with increasing numbers of training files; however the proportions of files in each of the categories were kept equal. Overall, the accuracies generally go up from 90% to up to a maximum of about 96% for approximately 180 files. A strange result is that the accuracy dramatically drops at 220 files for all algorithms before starting to go up again. This behaviour was strange considering that the numbers of files were increased by adding additional files to the existing set. This means that the existing vocabulary was retained and the change was the existing vocabulary plus the new vocabulary. Consequently, the feature set was enhanced rather than completely changed, making the drop in accuracy puzzling. To investigate this further, a different set of files were added to the existing set to make it up to 220. However, the behaviour was still apparent. This could be further investigated by going into the details of the individual feature words and their weights. However time did not permit this, and hence this was left as future work.

Another interesting result from Figure 4 is that the over fitting point for all the algorithms is at approximately 6060 files. At this point all the algorithms except Max Ent start to taper off in terms of accuracy and further ad hoc testing with larger number of training files gave even further drops in accuracy as expected over the fitting point.

Table 10. Testing accuracies over increasing training files

	Testing Accuracies based on selected Algorithms				
Increment of training files	NaiveBayes	Balanced Winnow	MCMaxEnt	MaxEnt	MaxEnL1
70	0.900	0.918	0.90	0.882	0.936
80	0.903	0.885	0.903	0.771	0.877
100	0.916	0.966	0.958	0.941	0.975
150	0.918	0.918	0.881	0.888	0.940
250	0.921	0.921	0.884	0.872	0.933
550	0.949	0.941	0.898	0.905	0.937
1050	0.935	0.972	0.925	0.948	0.950
3050	0.949	0.979	0.948	0.969	0.971
5050	0.972	0.990	0.974	0.977	0.980

Table 10 shows the testing accuracies for different algorithms with increasing numbers of training files from 70 through 5050.

The highest and lowest testing accuracies are shown over training files based on individual algorithms in the following Table 11. This information is taken from Table 10.

Table 11. Highest and lowest testing accuracies over increasing training files

Algorithms	Highest testing accuracy over increasing training files (no. of training files)		Lowest testing accuracy over increasing training files (no. of training files)	
Naïve Bayes	0.972	(5050)	0.900	(70)
Balanced Winnow	0.990	(5050)	0.885	(80)
MC MaxEnt	0.974	(5050)	0.881	(150)
MaxEnt	0.977	(5050)	0.771	(80)
MaxEnt L1	0.980	(5050)	0.877	(80)

Table 11 highlights that the highest accuracy of 0.990 was obtained over training file 5050 by Balanced Winnow algorithm. The lowest accuracy of 0.771 was obtained over training file 80 by Max Ent algorithm.

The data in Table 10 plotted in Figure 5 indicates the effect of the testing accuracies over increasing training files for imbalanced dataset.

Figure 5. Plot of testing accuracies versus the number of training files for imbalanced dataset

Figure 5 illustrates accuracies with increasing number of training files. In each of the categories, the proportions of the files were kept unequal. Like the previous graph, the accuracies have gone to a maximum of approximately 97% for 100 files after dropping down at 80 files. In this situation, the numbers of files were increased unequally and added to the existing set of files, which means new vocabulary was added to the existing vocabulary. An interesting result is that an over fitting point is noted for all the algorithms at approximately 5050 files, where the accuracy based on the algorithms start to taper off.

Table 12. Testing accuracies over increasing training files

	Testing Accuracies based on selected Algorithms				
Increment of training files	NaiveBayes	Balanced Winnow	MCMaxEnt	MaxEnt	MaxEnL1
70	0.927	0.927	0.891	0.864	0.918
80	0.877	0.929	0.868	0.885	0.903
100	0.933	0.950	0.858	0.858	0.908
150	0.918	0.918	0.933	0.874	0.925
250	0.915	0.909	0.921	0.854	0.909
550	0.960	0.945	0.952	0.925	0.956
1050	0.975	0.967	0.953	0.940	0.958
3050	0.978	0.977	0.954	0.957	0.967
5050	0.983	0.983	0.966	0.980	0.984

Table 12 shows the testing accuracies for different algorithms with increasing numbers of training files from 70 through 5050.

The highest and lowest accuracies are shown over training files based on individual algorithms in the following Table 13. This information is taken from Table 12.

Table 13. Highest and lowest testing accuracies over increasing training files

Algorithms	Highest testing accuracy over increasing training files (no. of training files)		Lowest testing accuracy over increasing training files (no. of training files)	
Naïve Bayes	0.983	(5050)	0.877	(80)
Balanced Winnow	0.983	(5050)	0.909	(250)
MC MaxEnt	0.966	(5050)	0.858	(100)
MaxEnt	0.980	(5050)	0.854	(250)
MaxEnt L1	0.984	(5050)	0.903	(100)

Table 13 highlights that the highest accuracy of 0.984 was obtained over training file 5050 by MaxEnt L1 algorithm. The lowest accuracy of 0.854 was obtained over training file 250 by MaxEnt algorithm.

The data in Table 12 plotted in Figure 6 indicates the effect of the testing accuracies over training files for imbalanced dataset.

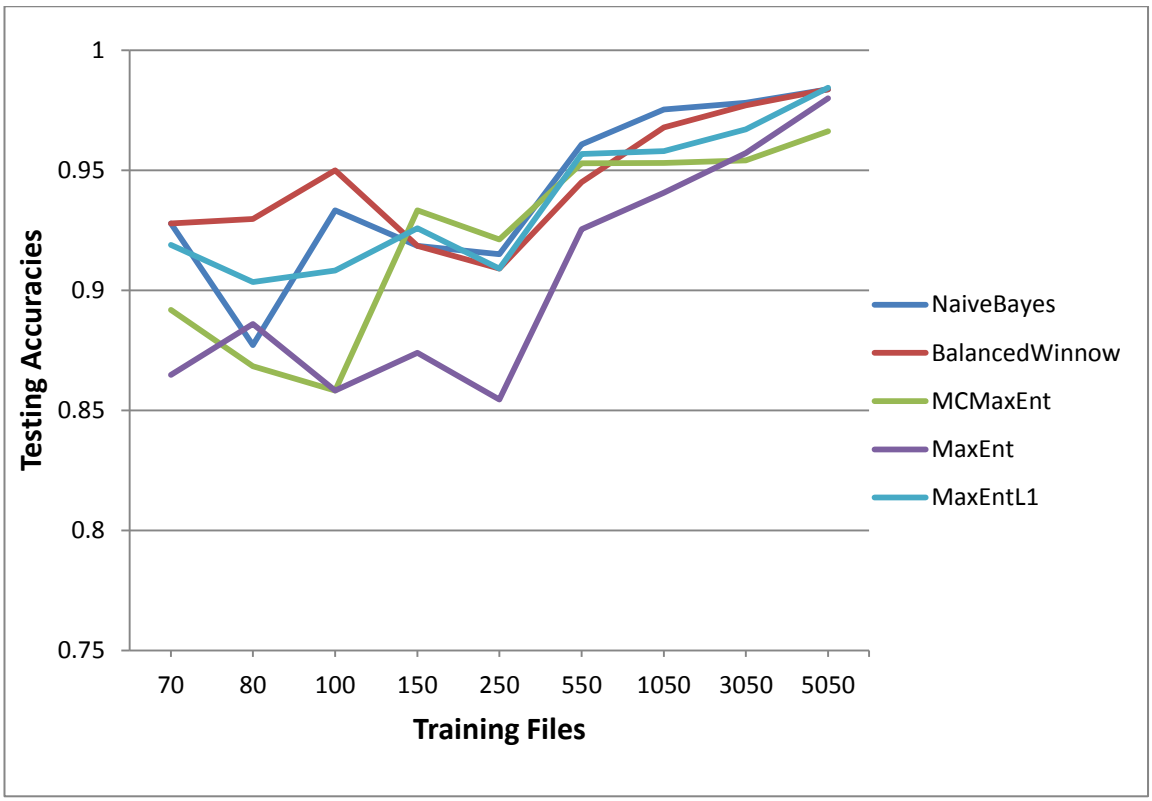

Figure 6. Plot of testing accuracies versus the number of training files for imbalanced dataset

Figure 6 illustrates accuracies with increasing numbers of training files. Like the previous graph, in each of the categories, the proportions of the files were kept unequal. In this situation, almost all the accuracies dramatically increase and drop at a certain number of files, and then finally goes up to taper where the accuracies seem to over fit at 5050 files. The exception to this is the algorithm MC Max Ent. This might be because of the addition of new vocabularies as well to the existing ones.

Overall, for all the experiments, different sets of files were added to the existing set to investigate further. Add on investigations can be done by going into the details of the individual feature words and weights.

4.1 Classification Performance Measures on Balanced Datasets

The categorization accuracy was measured with the help of Precision, Recall and F-value. Recall is defined as the proportion of documents correctly assigned to a category compared to all the documents in the category. Precision is the proportion of documents correctly assigned to a category from those that were attempted. F-measure combines the precision and recall value as a single figure to make more general observations instead of individual precision and recall value. For example, if “n” test documents and “m” classes are considered, then in this situation a classifier makes n *

m binary decisions. A two-by-two contingency table, called the Confusion Matrix, can be computed for each class.

	Truly YES	Truly NO
System YES	a	b
System NO	c	d

$\text{Therefore, Recall (R)} = \frac{a}{(a + c)}$; Where a + c > 0

$\text{Precision (P)} = \frac{a}{(a + b)}$; Where a + b > 0 and $\text{F} - \text{measure} = \frac{2(P * R)}{P + R}$

For balanced dataset, each increment of training data, precision, recall and F-measure were calculated with multiple algorithms which are tabulated in Table (D) in Appendix E. A summary of the mean value of precision, recall and F-measure were calculated with respect to algorithms on balanced dataset. These are shown in Table 14.

Table 14. Mean precision, recall and F-measure on implemented algorithms

Algorithms	Precision	Recall	F-measures
Naïve Bayes	0.965	0.934	0.947
Balanced Winnow	0.930	0.949	0.939
MC MaxEnt	0.922	0.952	0.936
MaxEnt	0.930	0.926	0.927
MaxEnt L1	0.951	0.949	0.949

The F-values are very close together (a variation of 0.022), suggesting very high precision together with a good scope for all of the algorithms. The best performing algorithm in terms of the F-value was Max EntL1 while the worst was Max Ent. The results affirm that Max EntL1 does surprisingly well in text classification tasks; hence it is used as a benchmark for text classification.

4.2 Classification Performance Measures on Imbalanced Datasets

For imbalanced datasets, each increment of training data, precision, recall and F-measure were calculated with multiple algorithms which are tabulated in Table (E) in Appendix E. A summary of the mean value of precision, recall and F-measure were calculated with respect to algorithms on imbalanced datasets. These are shown in Table 15.

Table 15. Mean precision, recall and F-measure on implemented algorithms

Algorithms	Precision	Recall	F-measures
Naïve Bayes	0.839	0.808	0.819
Balanced Winnow	0.853	0.932	0.889
MC MaxEnt	0.673	0.904	0.757
MaxEnt	0.773	0.855	0.804
MaxEnt L1	0.811	0.924	0.853

Like the previous table, the F-values in this table are very close together (a variation of 0.132) suggesting very high precision with good scope for all of the algorithms. The best performing algorithm in terms of the F-value was Balanced Winnow while the worst was MC Max Ent. The results show that Balanced Winnow performed well in text classification tasks; hence it is used as a benchmark for text classification.

For imbalanced datasets, each increment of training data, precision, recall and F-measure were calculated with multiple algorithms which were tabulated in Table (F) in Appendix E. A summary of the mean value of precision, recall and F-measure are calculated with respect to algorithms on imbalanced datasets. These are shown in Table 16.

Table 16. Mean precision, recall and F-measure on implemented algorithms

Algorithms	Precision	Recall	F-measures
Naïve Bayes	0.970	0.928	0.947
Balanced Winnow	0.947	0.958	0.951
MC MaxEnt	0.911	0.947	0.926
MaxEnt	0.926	0.908	0.915
MaxEnt L1	0.961	0.930	0.944

Like the previous table, the F-values in this table are very close together (a variation of 0.036) suggesting very high precision with a good scope for all of the algorithms. The best performing algorithm in terms of F-value was Balanced Winnow while the worst was Max Ent. The results show that Balanced Winnow performed well in text classification tasks; hence it is used as a benchmark for text classification.

Comparing Table 14, Table 15 and Table 16, Balanced Winnow performed best as the highest F-measure is 0.951.

Chapter 5. *Discussion*

The objective of this research was to investigate the impact of imbalanced data on common classifying algorithms. For this, Naïve Bayes, Balanced Winnow, and three variations of Max Ent (Max Ent, Max Ent L1and MC Max Ent) were used as implemented in MALLET. The input data was imported after the installation of MALLET. Four upper level topic codes of RCV1-v2 (edited version of Reuters Corpus) were used as the input data. The topic codes were CCAT, MCAT, ECAT and GCAT. Two categories CCAT and OTHERS were used in the balanced and imbalanced datasets. The OTHERS category contained a random mixture of MCAT, ECAT and GCAT. The balanced dataset and the imbalanced datasets had the same categories CCAT and OTHERS but had unequal proportions of files in each category. Three different experiments were carried using balanced and imbalanced datasets.

The results of the experiments carried out for the research are reported in Chapter 4. The discussion chapter contains an overall discussion of the results, including comparison with some results of other similar research.

From Chapter 4, it can be observed that Experiment (A) performed on balanced dataset yielded highest accuracy of 0.967 at training file size of 6060 with the help of the algorithm Max Ent L1. The best performing algorithm in this experiment in terms of F-value was Max Ent L1 as the highest F- measure was 0.949.

The corresponding highest accuracies on imbalanced datasets were 0.990 and 0.984 at training file size of 5050 from Experiments (B) and (C). The respective algorithms which aided to achieve the accuracies on imbalanced datasets were Balanced Winnow and Max Ent L1.

It is observed that the performance of the balanced dataset was less in terms of classification accuracy compared to the imbalanced datasets. The classification accuracies in this thesis were found to be different from other reported results of the reviews as the accuracies were based on different algorithms and different datasets having varied data distribution in the classification categories.

The best performing algorithm in Experiment (B) in terms of F-value was Balanced Winnow as the highest F-measure was 0.889. The best performing algorithm in Experiment (C) in terms of F-value was Balanced Winnow as the highest F-measure was 0.951. It can be seen that the classification accuracy and the F-value measure in terms of evaluation metrics are closely related.

Table 17 below ranks the best algorithms according to the performance on both balanced and imbalanced datasets.

Table 17. Ranking of Algorithms according to testing accuracies

Data Types	Highest Testing Accuracies	Best Algorithms ranked according to Highest Testing Accuracies
Imbalanced data (Experiment B)	0.990	Balanced Winnow
Imbalanced data (Experiment C)	0.984	MaxEnt L1
Balanced data (Experiment A)	0.967	MaxEnt L1

From the experiments done for this thesis, it can be seen that some of the algorithms performed better than the others. Although it is widely reported that Naïve Bayes has good classification accuracy on a variety of datasets, the results from this thesis show that other benchmarking algorithms performed better. The accuracy of classification is also dependent on the distribution of data in the classification categories. It was found that a training set having a different number of representatives from either class can result in a classifier that is subjective towards the majority class. However, when applied to a test set which is likely to be also imbalanced, the same classifier can yield positive accuracy. In an extreme case, the classifier can assign every single test case to the majority class; thereby an accuracy can be achieved that is equal to the proportion of the test cases having its place in the majority class. The balance on the training set can be restored either by under-sampling the large class or by oversampling the small class. This prevents bias towards a class by the classifier.

Only imbalanced datasets are considered for evaluation, as datasets in the real world tend to be imbalanced. Using Experiments (B) and (C) on imbalanced datasets, the highest testing accuracies are compared. From Table 17, it can be seen that Experiment (B), performed on an imbalanced dataset, yielded an extremely high accuracy of 0.990 using Balanced Winnow algorithm, compared to the rest of the accuracies.

Thus, Balanced Winnow can be considered as the best performing algorithm compared to the other algorithms, namely Naïve Bayes, Max Ent, Max Ent L1 and MC Max Ent, used in the experiments of this research in terms of F-measure and testing accuracies.

An accuracy value of 0.990 was achieved on an imbalanced dataset (Experiment B) using Balanced Winnow algorithm where one of the two categories, CCAT was kept constant and OTHERS was incremented. The highest accuracy of 0.984 was achieved on an imbalanced dataset (Experiment C) using Max Ent L1 algorithm where one of the two categories, OTHERS was kept constant and CCAT was incremented. A total of 26,100 articles were used for Experiments (B) and (C). Comparing these results to, Krishnalal et al. (2010), the authors of this work report accuracies of 83.25% 80.22%, and 82.26 % for the major categories such as sports, finance and politics using *kNN*. Additionally, accuracies of 87.67%, 82.57%, and 86.55% were found using SVM; however much higher values of 92.45%, 96.34%, and 90.76% were found to be using the combined model HMM-SVM. The database for these experiments was also newspaper articles from online newspapers New Indian Express, Times of India, The Economic Times and Business Line.

Ting et al. (July 2011) achieved accuracies of 96. 9 % and 95.5% respectively on non-pre-processed and pre-processed data using Naïve Bayes classifier based model built in WEKA. The database contained 4000 documents classified into four categories - business, politics, sports and travel. The classifying algorithms used were decision tree (DT), Support Vector Machines (SVM), neural networks and Naïve Bayes using WEKA. The objective of the experimental evaluation was twofold. Firstly, the experiment investigated whether the pre-processing phase was useful to infer improved classification accuracy and performance when compared with the non-pre-processed data. Secondly, it compared the classification accuracy when different classifiers were applied.

The similarity between the experiments for this research thesis and the other experiments conducted for the reviews stated above was that all the work tried to investigate the performance of the common classifying algorithms on certain types of datasets. However, the present research tries to investigate the effect of imbalanced data on document classification algorithms. Relating the two previous research studies to the present research, it can be seen that the accuracies in this research are very much consistent or higher compared to the discussed researches. However the data size used

for this research was substantially larger; hence the results would be more generalised. The algorithms used in this research were different compared to the discussed research studies. However as the results show, the performances achieved are very much similar.

The accuracies for this research thesis are comparable with the reviews. The implemented algorithms in tool MALLET are different to the ones used in the reviews; nevertheless, a comparison of the performance can be done across algorithms.

Chapter 6. *Conclusion and Future Recommendations*

This thesis examined the classification performance or the testing accuracies of common classifying algorithms on increasing the amount of training data. The tool used for the experiments was MALLET, which has implementation of several classification algorithms. The main focus of the experiments was to investigate whether imbalanced data made any impact on the accuracy of classifications as this is typically the situation in real life, where there are far fewer documents in a classification category compared to all other documents.

It was found that imbalanced datasets had an impact opposite to the impact reported in previous works, albeit with different set of algorithms. The classification accuracies were found to be higher for the imbalanced datasets compared to the balanced sets for almost all the algorithms used in the experiments using MALLET. The accuracies were found to be generally higher for the algorithms used for this research both for balanced as well as imbalanced datasets. For example, the accuracies were found to be up to 90.76% in Krishnalal et al. (2010) and 96.9 % in Ting et al. (July 2011) whereas the accuracy was found to be up to 99.0% in this thesis. Considering imbalanced datasets in this research, the highest testing accuracy derived from Table 11 in Chapter 4 (Experiment B) on the imbalanced dataset was 0.990 using Balanced Winnow. The highest testing accuracy derived from Table 13 in Chapter 4 (Experiment C) on imbalanced dataset was 0.984 using Max Ent L1. Out of these accuracies, the highest accuracy was 0.990 obtained through Balanced Winnow. This is an impressive result obtained on the imbalanced data compared to other reported results for imbalanced data.

It was observed that the F-values for all the experiments were very close together with minute variations suggesting high rates of precision for all the algorithms. The F-value measures indicated that the Balanced Winnow algorithm performed better compared to other benchmarking algorithms. The highest F-measure was 0.951 for the Balanced Winnow algorithm. With respect to the research objective, it was found that the imbalance in the dataset does make a difference in the accuracy; however, in this

research it was for the better. The accuracy with respect to the imbalanced data was better than with the balanced data. Due to the probabilistic nature of the algorithms used for testing, it is difficult and time consuming to ascertain the reason for this. This could be due to the vocabulary statistics of the documents from RCV1 or, even more interestingly, related to the parameters of the algorithms used. Due to time constraints, this issue could not be further investigated and is left as future research. Another odd observation from Figure 5 and Figure 6 is that a sudden drop in accuracy occurs at 80 and 250 training files respectively for all the algorithms used in the tests. This suggests a phenomenon pertinent to the data rather than the algorithms. Although this was investigated, no conclusion could be drawn due to time constraints, and is left for future research as well.

References

Al-Mubaid, H., & Umair, S. A. (2006). A new text categorization technique using Distributional Clustering and Learning Logic. *IEEE Transactions on Knowledge and Data Engineering, 18*(9), 1156-1165.

Alchemy API, Inc. (2013). Retrieved from http://www.alchemyapi.com/api/text-categorization/

The Apache Software Foundation. (2014). Retrieved from http://mahout.apache.org/

Bao, Y., & Ishii, N. (2002). Combining Multiple K-Nearest Neighbor Classifiers for Text Classification by Reducts Symposium conducted at the meeting of the Discovery Science, 5th International Conference, Liibeck, Germany.

Beuls, K., Pflugfelder, B., & Hanbury, A. (2010). Comparative analysis of Balanced Winnow and SVM in large scale patent categorization Symposium conducted at the meeting of the 10th Dutch-Belgian Information Retrieval Workshop, Nijmegen.

Bi, Y., Bell, D., Wang, H., Guo, G., & Greer, K. (2004). Combining Multiple Classifiers Using Dempster's Rule of Combination for Text categorization. *Springer Link*, 127-138.

Bratu, C. V., Muresan, T., & Potolea, R. (2008, 28-30 Aug. 2008). *Improving Classification Accuracy through Feature Selection*. presented at the meeting of the 4th International conference on Intelligent Computer Communication and Processing, Cluj-Napoca

Chawla, N. V., Japkowicz, N., & Kołcz, A. R. (June 2004). Editorial: special issue on learning from imbalanced data sets. *ACM SIGKDD Explorations Newsletter 6*(1), 1-6.

Cho, S.-B., & Lee, J.-H. (2003). Learning Neural Network Ensemble for Practical Text Classification. *Springer Link, 2690*, 1032-1036.

Dash, M., & Liu, H. (1997). Feature Selection for Classification. *Intelligent Data Analysis, 1*(1-4), 131-156.

Ganganwar, V. (April 2012). An overview of classification algorithms for imbalanced datasets. *International Journal of Emerging Technology and Advanced Engineering, 2*(4).

Harrag, F., El-Qawasmeh, E., & Pichappan, P. (2009, 28-31 July 2009). *Improving Arabic text categorization using decision trees*. presented at the meeting of the Networked Digital Technologies, Ostrava

Hassan, S., Rafi, M., & Shaikh, M. S. (2012). Comparing SVM and NaIve Bayes Classifiers for text categorization with Wikitology as knowledge enrichment. *Multitopic Conference (INMIC), 2011 IEEE 14th International*.

Joshi, M. V. (2002). *Learning classifier models for predicting rare phenomena [Ph.D.Thesis]*

Kamruzzaman, S. M. (2012). Text classification using Artificial Intelligence. *Journal of Electrical Engineering, The Institution of Engineers, Bangladesh, EE33*(I & II).

Khan, A., Baharudin, B., Lee, L. H., & Khan, K. (2010). A review of Machine Learning Algorithms for text-documents classification. *Journal of Advances in Information Technology, 1*(1), 4-20.

Koster, C. H. A., Seutter, M., & Beney, J. (2003). Classifying patent applications with Winnow. *University of Nijmegen,The Netherlands*.

Krishna Veni, C. V., & Sobha Rani, T. (2011). On the classification of imbalanced datasets. *IJCST(1,2Dept. of Computers and Information Sciences University of Hyderabad, India), 2*(SP 1).

Krishnalal, G., Rengarajan, S. B., & Srinivasagan, K. G. (2010). A new text mining approach based on HMM-SVM for Web News Classification. *International Journal of Computer Applications 1*(19), 98-104.

Lewis, D. D., Yang, Y., Rose, T. G., & Li, F. (2004). RCV1: A new benchmark collectionfor text categorization research. *Journal of Machine Learning Research, 5*, 361-397.

Li, Y., & Chen, C. (2012, 29-31 May 2012). *Research on the feature selection techniques used in text classification*. presented at the meeting of the 9th International Conference on Fuzzy Systems and Knowledge Discovery, Sichuan

Li, Y., Sun, G., & Zhu, Y. (2010, 15-17 Oct. 2010). *Data imbalance problem in text classification*. presented at the meeting of the Third International Symposium on Information Processing, Qingdao

Manning, C. D., Raghavan, P., & Schiitze, H. (2008). *Introduction to Information Retrieval*: Cambridge University Press,USA.

Manning, C. D., & Schiitze, H. (1999). *Foundations of Statistical Natural Language Processing* London, England: The MIT Press Cambridge, Massachusetts

McCallum, A. (2013). *MALLET: A Machine Learning for Language Toolkit*. Retrieved from http://mallet.cs.umass.edu/index.php

McCallum, A., & Nigam, K. (1998). A comparison of event models for Naive Bayes text classication. *98 workshop on learning for text categorization*.

Nigam, K., Lafferty, J., & McCallum, A. (1999). Using Maximum Entropy for text classification. *IJCAI-99 Workshop on Machine Learning for Information Filtering*, 61-67.

Olson, D. L. (2005). Data set balancing. In *Data Mining and Knowledge Management* (pp. 71-80): Springer Berlin Heidelberg.

Ramdass, D., & Seshasai, S. (2009). Document classification for newspaper articles.

Rose, T., Stevenson, M., & Whitehead, M. (2002). *The Reuters Corpus Volume 1: From yesterday's news to tomorrow's language resources* Symposium conducted at the meeting of the Proceedings of the Third International Conference on Language Resources and Evaluation

Ruiz, M. E., & Srinivasan, P. (1999). Hierarchical Neural Networks for Text Categorization *22 nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 281-282.

Sebastiani, F. (2005). Text categorization. *Department of Pure and Applied Mathematics,University of Padova,Italy*.

Sun, Y., Kamel, M. S., Wong, A. K. C., & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *The Journal of Pattern Recognition Society,Science Direct, 40*, 3358-3378.

Thornton, C. (n.d.). *Machine Learning - Lecture 4: The Naive Bayes Classifier*.

Ting, S. L., Ip, W. H., & Tsang, A. H. C. (July 2011). Is Naïve Bayes a good classifier for document classification? *International Journal of Software Engineering and Its Applications, 5*(3).

Vajda, S. a., & Fink, G. A. (2010, 16-18 Nov 2010). Strategies for training robust neural network based digit recognizers on unbalanced data set *IEEE.* Symposium conducted at the meeting of the 12th International Conference on Frontiers in Handwriting Recognition, Kolkata.

Vert, J.-P. (July 2001). Adaptive context trees and text clustering. *Dept. of Math. & Applications, Ecole Normale Superieure, Paris, France, 47*(5), 1884-1901

Wang, S., & Yao, X. (August 2012). Multiclass imbalance problems: Analysis and potential solutions. *IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics, 42*(4), 1119-1130.

Weiss, G. (2004). Mining with rarity: A unifying framework. *SIGKDD Explorations Special Issue on Learning from Imbalanced Datasets, 6*(1), 7-19.

Zhu, S., Ji, X., Xu, W., & Gong, Y. (2005). Multilabelled classification using Maximum Entropy method*ACM.* Symposium conducted at the meeting of the Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval New York,USA.

Appendix A. Topic Codes

Code	Description
1POL	Current News - Politics
2ECO	Current News - Economics
3SPO	Current News - Sport
4GEN	Current News - General
6INS	Current News - Insurance
7RSK	Current News - Risk News
8YDB	Temporary
9BNX	Temporary
ADS10	Current News - Advertising
BNW14	Current News - Business News
BRP11	Current News - Brands
C11	Strategy/Plans
C12	Legal/Judicial
C13	Regulation/Policy
C14	Share Listings
C15	Performance
C151	Accounts/Earnings
C1511	Annual Results
C152	Comment/Forecasts
C16	Insolvency/Liquidity
C17	Funding/Capital
C171	Share Capital
C172	Bonds/Debt Issues
C173	Loans/Credits
C174	Credit Ratings
C18	Ownership Changes
C181	Mergers/Acquisitions
C182	Asset Transfers
C183	Privatisations
C21	Production/Services
C22	New Products/Services
C23	Research/Development
C24	Capacity/Facilities
C31	Markets/Marketing
C311	Domestic Markets
C312	External Markets
C313	Market Share
C32	Advertising/Promotion
C33	Contracts/Orders
C331	Defence Contracts
C34	Monopolies/Competition
C41	Management
C411	Management Moves
C42	Labour
CCAT	Corporate/Industrial
E11	Economic Performance
E12	Monetary/Economic
E121	Money Supply
E13	Inflation/Prices
E131	Consumer Prices
E132	Wholesale Prices
E14	Consumer Finance
E141	Personal Income
E142	Consumer Credit
E143	Retail Sales
E21	Government Finance
E211	Expenditure/Revenue
E212	Government Borrowing
E31	Output/Capacity
E311	Industrial Production
E312	Capacity Utilization
E313	Inventories
E41	Employment/Labour
E411	Unemployment
E51	Trade/Reserves
E511	Balance Of Payments
E512	Merchandise Trade
E513	Reserves
E61	Housing Starts
E71	Leading Indicators
ECAT	Economics
ENT12	Current News - Entertainment
G11	Social Affairs
G111	Health/Safety
G112	Social Security
G113	Education/Research
G12	Internal Politics
G13	International Relations
G131	Defence
G14	Environment
G15	European Community
G151	Ec Internal Market
G152	Ec Corporate Policy
G153	Ec Agriculture Policy
G154	Ec Monetary/Economic
G155	Ec Institutions
G156	Ec Environment Issues
G157	Ec Competition/Subsidy
G158	Ec External Relations
G159	Ec General
GCAT	Government/Social
GCRIM	Crime, Law Enforcement
GDEF	Defence
GDIP	International Relations
GDIS	Disasters And Accidents
GEDU	Education
GENT	Arts, Culture, Entertainment
GENV	Environment And Natural World
GFAS	Fashion
GHEA	Health
GJOB	Labour Issues
GMIL	Millennium Issues
GOBIT	Obituaries
GODD	Human Interest
GPOL	Domestic Politics
GPRO	Biographies, Personalities, People
GREL	Religion
GSCI	Science And Technology
GSPO	Sports
GTOUR	Travel And Tourism
GVIO	War, Civil War
GVOTE	Elections
GWEA	Weather
GWELF	Welfare, Social Services
M11	Equity Markets
M12	Bond Markets
M13	Money Markets
M131	Interbank Markets
M132	Forex Markets
M14	Commodity Markets
M141	Soft Commodities
M142	Metals Trading
M143	Energy Markets
MCAT	Markets
MEUR	Euro Currency
PRB13	Current News - Press Release Wires

Appendix B. XML Format of a News Document

<?xml version="1.0" encoding="ISO-8859-1"?>

-<newsitem xml:lang="en" date="1996-08-20" id="root" itemid="2286"> <title>MEXICO: Recovery excitement brings Mexican markets to life.</title> <headline>Recovery excitement brings Mexican markets to life.</headline> <byline>Henry Tricks</byline> <dateline>MEXICO CITY</dateline>

- Emerging evidence that Mexico's economy was back on the recovery track sent Mexican markets into a buzz of excitement Tuesday, with stocks closing at record highs and interest rates at 19-month lows.

"Mexico has been trying to stage a recovery since the beginning of this year and it's always been getting ahead of itself in terms of fundamentals," said Matthew Hickman of Lehman Brothers in New York. tes at 19-month lows.

"Now we're at the point where the fundamentals are with us. The history is now falling out of view."

That history is one etched into the minds of all investors in Mexico: an economy in crisis since December 1994, a free-falling peso and stubbornly high interest rates.

This week, however, second-quarter gross domestic product was reported up 7.2 percent, much stronger than most analysts had expected. Interest rates on governent Treasury bills, or Cetes, in the secondary market fell on Tuesday to 23.90 percent, their lowest level since Jan. 25, 1995.

The stock market's main price index rallied 77.12 points, or 2.32 percent, to a record 3,401.79 points, with volume at a frenzied 159.89 million shares.

Confounding all expectations has been the strength of the peso, which ended higher in its longer-term contracts on Tuesday despite the secondary Cetes drop and expectations of lower benchmark rates in Tuesday's weekly auction.

With U.S. long-term interest rates expected to remain steady after the Federal Reserve refrained from raising short-term rates on Tuesday, the attraction of Mexico, analysts say, is that it offers robust returns for foreigners and growing confidence that they will not fall victim to a crumbling peso.

"The focus is back on Mexican fundamentals," said Lars Schonander, head of researcher at Santander in Mexico City. "You have a continuing decline in inflation, a stronger-than-expected GDP growth figure and the lack of any upward move in U.S. rates."

Other factors were also at play, said Felix Boni, head of research at James Capel in Mexico City, such as positive technicals and economic uncertainty in Argentina, which has put it and neighbouring Brazil's markets at risk.

"There's a movement out of South American markets into Mexico," he said. But Boni was also wary of what he said could be "a lot of hype."

The economic recovery was still export-led, and evidence was patchy that the domestic consumer was back with a vengeance. Also, corporate earnings need to grow strongly to justify the run-up in the stock market, he said.

</text>

<copyright>(c) Reuters Limited 1996</copyright>

-<metadata>

-<codes class="bip:countries:1.0">

-<code code="MEX">

<editdetail date="1996-08-20" action="confirmed" attribution="Reuters BIP Coding Group"/>

</code>

</codes>

-<codes class="bip:topics:1.0">

-<code code="E11">

<editdetail date="1996-08-20" action="confirmed" attribution="Reuters BIP Coding Group"/>

```
</code>
-<code code="ECAT">
 <editdetail date="1996-08-20" action="confirmed" attribution="Reuters BIP Coding Group"/>
 </code>
 -<code code="M11">
 <editdetail date="1996-08-20" action="confirmed" attribution="Reuters BIP Coding Group"/>
 </code>
 -<code code="M12">
<editdetail date="1996-08-20" action="confirmed" attribution="Reuters BIP Coding Group"/>
 </code>
 -<code code="MCAT">
 <editdetail date="1996-08-20" action="confirmed" attribution="Reuters BIP Coding Group"/>
 </code>
</codes>
 <dc value="Reuters Holdings Plc" element="dc.publisher"/>
 <dc value="1996-08-20" element="dc.date.published"/>
 <dc value="Reuters" element="dc.source"/>
 <dc value="MEXICO CITY" element="dc.creator.location"/>
 <dc value="MEXICO" element="dc.creator.location.country.name"/>
 <dc value="Reuters" element="dc.source"/>
 </metadata> c
</newsitem>
```

Appendix C. Text Format of a News Document

Chrysler Corp. Tuesday announced $380 million in new investments for South America, including assembly plants for pickup trucks and diesel engines in Brazil and the expansion of a Jeep plant now being built in Argentina.

Chrysler, which is cautiously trying to rebuild its international presence, said the projects in Brazil were worth about $315 million, and the expansion in Argentina was worth about $65 million.

Roughly one third of the total investment, or about $126.6 million, will come from Chrysler's suppliers, who will play a major role in the automaker's low-risk global growth strategy, Chrysler Chairman Robert Eaton said.

"We don't intend to make risky investments just to be a major player in emerging markets," added Thomas Gale, Chrysler's executive vice president of international operations. "We're quite content to grow at a steady pace in regions where we see solid opportunities."

Eaton said the investments will boost Chrysler's sales in the Mercosur free-trade zone, which groups Argentina, Brazil, Paraguay and Uruguay.

But the company's limited production capacity will allow it only to grab a small portion of the Mercosur market away from rivals General Motors Corp., Ford Motor Co. and Volkswagen AG, he said.

"We are targeting very specific market segments," he said. "We don't have any interest or desire to offer a vehicle for every possible application."

The new Brazilian plant, which will be Chrysler's third limited-production facility in South America, will assemble the automaker's all-new Dakota compact pickup truck for sale in Argentina, Brazil, Paraguay and Uruguay, the countries in the Mercosur free-trade zone.

In Argentina, Chrysler said it will add production of about 6,000 Jeep Cherokees a year at a plant now under construction in the Cordoba province. The plant is already scheduled to build about 14,000 Jeep Grand Cherokees per year starting next April, and Cherokee output will begin in 1998.

A site for the Brazilian plant will be selected by year-end and vehicles will roll off the assembly line starting in mid-1998, Chrysler said. Production, however, will be modest, with 12,000 trucks in the first year and an ultimate capacity of 40,000 units annually. Employment will start at 400 people.

The trucks at first will be largely assembled from "complete knock-down" kits shipped from the United States, but the automaker intends to meet the Mercosur trade bloc's 60 percent local content requirement after three years.

Chrysler has not decided whether to market the Dakota under the Dodge brand name or under one of its other brands. The automaker now uses the only Jeep and Chrysler brand names outside the United States, Canada and Mexico.

The $315 million Brazil investment also includes a new diesel engine plant to be built by Detroit Diesel Corp..

The $10 million facility will supply the company's Italian-designed VM Motori four-cylinder turbocharged diesel engines for use in the Brazilian Dakota as well as in Jeep models built in Argentina. Chrysler installs about 40,000 of the engines annually into minivans and Jeep Grand Cherokees sold in Europe.

Others suppliers supporting the Chrysler by opening plants in Latin America include Dana Corp., Johnson Controls Inc., Lear Corp., Lear Corp. United Technologies Corp. and PPG Industries Inc., Chrysler executives said.

Eaton said total annual vehicle sales in the four-country Mercosur region will increase from about 2 million units currently to about 2.5 million by the end of the decade.

"We think this is a major growth area," Eaton said. "It's politically and economically a stable region, we think with particularly rising consumer buying power."

Including a small plant in Venezuela that assembles Cherokees and Neon small cars from kits, the investments announced Tuesday bring to $735 million the total financial commitments Chrysler and its suppliers have made in South America, the company said.

Chrysler stock rose 25 cents to close at $28.875 Tuesday on the New York Stock Exchange.

Appendix D. RCV1 File Lists Used for the Experiments

```
{
public static boolean separateIlists = false;
public static double trainingProportion = 0.6;
public static double validationProportion = 0.1;
public static Random r = new Random ();
public static String stopWordListFile  = "stoplists/en.txt";
//Input data files directory.
public static String inputDirectory = "../mdata/equal_data/data_6100";
//public static String inputDirectory = "../mdata/equal_data/data_6060";
//public static String inputDirectory = "../mdata/equal_data/data_6020";
//public static String inputDirectory = "../mdata/equal_data/data_6000";
//public static String inputDirectory = "../mdata/equal_data/data_4000";
//public static String inputDirectory = "../mdata/equal_data/data_2000";
//public static String inputDirectory = "../mdata/equal_data/data_260";
//public static String inputDirectory = "../mdata/equal_data/data_220";
//public static String inputDirectory = "../mdata/equal_data/data_180";
//public static String inputDirectory = "../mdata/equal_data/data_140";
//public static String inputDirectory = "../mdata/equal_data/data_100";
//public static String inputDirectory = "../mdata/equal_data/data_60";
//public static String inputDirectory = "../mdata/equal_data/data_20";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT5000";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT3000";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT1000";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT500";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT200";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT100";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT50";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT30";
//public static String inputDirectory = "../mdata/Training_Others_50/data_CCAT20";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others5000";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others3000";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others1000";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others500";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others200";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others100";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others50";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others30";
//public static String inputDirectory = "../mdata/Training_CCAT_50/data_Others20";
//public static String inputDirectory = "../mdata/Training_Others_50_old/data_CCAT3000";
//public static String inputDirectory = "../mdata/LargeTrSet";
//public static String inputDirectory = "../mdata/data50Tr";
//public static String inputDirectory = "../mdata/dataSmall2";
//public static String inputDirectory = "../mdata/sample-data/web";
        static public void main (String[] args)
        {
//LogManager.getLogManager().getLogger(Logger.GLOBAL_LOGGER_NAME).setLevel(Level.FINEST);
multipleTrainers();
}
```

```
private static void multipleTrainers() {
InstanceList ilistTr;
InstanceList ilistTe;
InstanceList ilistVa;
if (!separateIlists) {
//split the data into training, validation and test.
InstanceList ilist = createInstances(inputDirectory + "/RandomSplit");
//This instanceList splitter is bugged. It does not split the alphabets.
//InstanceList[] ilists = ilist.split (r, new double[] {trainingProportion,1-trainingProportion-validationProportion,validationProportion});
InstanceList[] ilists = ilist.split (new Random(1000), new double[] {trainingProportion,1-trainingProportion-validationProportion,validationProportion});
ilistTr = ilists[0];
ilistTe = ilists[1];
ilistVa = ilists[2];
} else {
ilistTr = createInstances(inputDirectory + "/Tr");
ilistTe = createInstances(inputDirectory + "/Te");
ilistVa = createInstances(inputDirectory + "/Va");
}
ArrayList<ClassifierTrainer> trainers = new ArrayList<ClassifierTrainer>();
trainers.add(new NaiveBayesTrainer());
trainers.add(new NaiveBayesEMTrainer());
trainers.add(new BalancedWinnowTrainer());
trainers.add(new C45Trainer());
trainers.add(new DecisionTreeTrainer());
trainers.add(new MaxEntTrainer());
trainers.add(new MaxEntL1Trainer());
trainers.add(new MCMaxEntTrainer());
//trainers.add(new WinnowTrainer());
//trainers.add(new MaxEntGETrainer());
//trainers.add(new MaxEntGERangeTrainer());
//trainers.add(new MaxEntPRTrainer());
ArrayList<Classifier> classifiers = new ArrayList<Classifier>();
for (ClassifierTrainer ct : trainers) {
classifiers.add(ct.train(ilistTr));
}
```

Appendix E. Additional Tables

Table (A). Training, testing and validation alphabet size

Input Imbalanced data (Unequal Increment of Training Files for CCAT and OTHERS)		Training/ Testing/ Validation alphabet size
CCAT	OTHERS	
50	20	8901
50	30	9141
50	50	9421
50	100	10390
50	200	11839
50	500	14833
50	1000	18227
50	3000	29352
50	5000	36753

Table (B). Training, testing and validation alphabet size

Input Imbalanced data (Unequal Increment of Training Files for CCAT and OTHERS)		Training/ Testing/ Validation alphabet size
CCAT	OTHERS	
50	20	9319
50	30	9340
50	50	9421
50	100	9630
50	200	10130
50	500	12042
50	1000	14544
50	3000	20996
50	5000	26262

Table (C). Training, testing and validation files

Input Imbalanced data (Increment of Training Files for CCAT and OTHERS)		Total Training Files	Total Validation Files	Total Testing Files
OTHERS	CCAT			
50	20	222	37	111
50	30	228	38	114
50	50	240	40	120
50	100	270	45	135
50	200	330	55	165
50	500	510	85	255
50	1000	810	135	405
50	3000	2010	335	1005
50	5000	3210	535	1605

Table (D). The overall performances of the implemented system on training data

Algorithms	Naïve Bayes			Balanced Winnow			MC MaxEnt			MaxEnt			MaxEnt L1		
Balanced datasets	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure
20	0.958	0.945	0.951	0.889	0.937	0.911	0.899	0.937	0.917	0.908	0.895	0.901	0.926	0.939	0.932
60	0.963	0.910	0.935	0.927	0.944	0.935	0.918	0.939	0.928	0.914	0.889	0.901	0.945	0.916	0.93
100	0.968	0.942	0.954	0.927	0.949	0.937	0.927	0.932	0.928	0.932	0.928	0.929	0.959	0.946	0.952
140	0.974	0.926	0.948	0.952	0.936	0.943	0.914	0.963	0.937	0.944	0.917	0.93	0.948	0.948	0.947
180	0.965	0.938	0.951	0.927	0.956	0.941	0.880	0.958	0.896	0.927	0.912	0.918	0.965	0.957	0.96
220	0.968	0.913	0.939	0.912	0.916	0.913	0.896	0.961	0.927	0.892	0.910	0.900	0.948	0.948	0.947
260	0.949	0.925	0.936	0.902	0.959	0.929	0.894	0.950	0.920	0.902	0.902	0.901	0.932	0.936	0.933
2000	0.962	0.939	0.950	0.927	0.959	0.942	0.931	0.957	0.943	0.927	0.94	0.933	0.939	0.96	0.948
4000	0.967	0.949	0.957	0.943	0.952	0.947	0.939	0.966	0.952	0.925	0.943	0.933	0.948	0.953	0.95
6000	0.963	0.932	0.947	0.952	0.959	0.954	0.939	0.946	0.942	0.954	0.945	0.949	0.957	0.954	0.954
6020	0.964	0.939	0.951	0.942	0.955	0.948	0.948	0.961	0.954	0.958	0.954	0.955	0.962	0.959	0.96
6060	0.975	0.943	0.949	0.950	0.967	0.958	0.955	0.957	0.955	0.951	0.953	0.951	0.968	0.967	0.967
6100	0.970	0.942	0.955	0.946	0.958	0.951	0.955	0.958	0.956	0.966	0.952	0.958	0.966	0.963	0.964

Table (E). The overall performances of the implemented system on training data

Algorithms	Naïve Bayes			Balanced Winnow			MC MaxEnt			MaxEnt			MaxEnt L1		
Imbalanced datasets	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure
70	0.985	0.873	0.925	0.900	0.969	0.933	0.871	0.968	0.916	0.900	0.913	0.906	0.971	0.931	0.950
80	0.964	0.859	0.908	0.929	0.854	0.889	0.912	0.896	0.903	0.894	0.718	0.795	0.947	0.830	0.884
100	0.964	0.870	0.914	0.946	0.981	0.963	0.928	0.981	0.953	0.946	0.929	0.937	0.982	0.964	0.972
150	0.936	0.893	0.913	0.888	0.933	0.909	0.761	0.979	0.856	0.920	0.852	0.884	0.968	0.910	0.937
250	0.961	0.819	0.884	0.826	0.914	0.867	0.730	0.883	0.799	0.769	0.816	0.791	0.865	0.918	0.890
550	0.868	0.913	0.889	0.786	0.960	0.864	0.606	0.948	0.738	0.704	0.877	0.780	0.737	1	0.848
1050	0.814	0.733	0.771	0.833	0.957	0.890	0.518	0.875	0.650	0.685	0.902	0.778	0.685	0.925	0.786
3050	0.575	0.622	0.597	0.803	0.868	0.834	0.378	0.694	0.489	0.590	0.906	0.714	0.606	0.930	0.733
5050	0.491	0.697	0.576	0.770	0.959	0.853	0.360	0.916	0.516	0.557	0.790	0.653	0.540	0.916	0.679

Table (F). The overall performances of the implemented system on training data

Algorithms	Naïve Bayes			Balanced Winnow			MC MaxEnt			MaxEnt			MaxEnt L1		
Imbalanced datasets	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure	Precision	Recall	F-measure
70	1	0.862	0.925	0.928	0.919	0.923	0.880	0.880	0.879	0.900	0.818	0.856	0.980	0.859	0.915
80	0.872	0.872	0.871	0.872	0.979	0.922	0.763	0.954	0.847	0.800	0.956	0.870	0.854	0.940	0.894
100	0.949	0.918	0.933	0.915	0.981	0.946	0.762	0.937	0.839	0.830	0.875	0.851	0.898	0.913	0.905
150	0.962	0.904	0.931	0.949	0.914	0.93	0.911	0.972	0.939	0.924	0.869	0.895	0.962	0.915	0.937
250	1	0.875	0.933	0.928	0.919	0.923	0.918	0.947	0.931	0.938	0.836	0.883	0.979	0.880	0.926
550	0.975	0.975	0.974	0.951	0.979	0.964	0.980	0.961	0.970	0.985	0.922	0.952	0.995	0.953	0.973
1050	0.991	0.979	0.984	0.985	0.977	0.980	1	0.947	0.972	0.976	0.954	0.964	0.991	0.960	0.974
3050	0.991	0.985	0.987	0.994	0.981	0.987	0.997	0.955	0.975	0.933	0.961	0.976	0.998	0.967	0.982
5050	0.994	0.989	0.991	0.993	0.989	0.990	0.995	0.970	0.982	0.996	0.982	0.988	0.999	0.984	0.991