ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES

MSc THESIS

Özge AKDOĞAN

EFFECTS OF FEATURE EXTRACTION TECHNIQUES ON CLASSIFICATION OF TURKISH TEXTS

DEPARTMENT OF COMPUTER ENGINEERING

ADANA-2018

ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES

EFFECTS OF FEATURE EXTRACTION TECHNIQUES ON CLASSIFICATION OF TURKISH TEXTS

Özge AKDOĞAN

MSc THESIS

DEPARTMENT OF COMPUTER ENGINEERING

We certify that the thesis titled above was received and approved the award of degree of the Master of Science by the board of jury on

……..…	……..……	…..……..……..…….
Prof. Dr. Selma Ayşe ÖZEL	Assoc. Prof. Dr. Zekeriya TÜFEKÇİ	Assist. Prof. Dr. Alper K. DEMİR
SUPERVISOR	MEMBER	MEMBER

This MSc Thesis is written at the Computer Engineering Department of Institute of Natural and Applied Sciences of Çukurova University.
Registration Number:

Prof. Dr. Mustafa GÖK
Director
Institude of Natural and Applied Science

ABSTRACT

MSc THESIS

EFFECTS OF FEATURE EXTRACTION TECHNIQUES ON CLASSIFICATION OF TURKISH TEXTS

ÖZGE AKDOĞAN

ÇUKUROVA UNIVERSITY
INSTITUTE OF NATURAL AND APPLIED SCIENCES
DEPARTMENT OF COMPUTER ENGINEERING

Supervisor : Prof. Dr. Selma Ayşe ÖZEL
Year: 2018, Pages: 81
Juries : Prof. Dr. Selma Ayşe ÖZEL
: Assoc. Prof. Dr. Zekeriya TÜFEKÇİ
: Asst. Prof. Dr. Alper Kamil DEMİR

The purpose of this thesis is to determine the most effective method for extracting features and to develop an effective method of selecting features for the classification of Turkish documents in different types. We analyze the effects of preprocessing methods, weighting schemes, and feature selection on the performance of Turkish document classification. In the study, 5 different term weighting methods that are *tf, tp, logtf, normtf, tf*idf* are compared and it is found that "*tf*" and "*tf*idf*" give the best results. After that effects of stopwords removal are investigated, and it is observed that stopwords removal improves classification performance. Then we compare 5 different stemming algorithms that are Zemberek, Affix Stripping, Fixed Prefix 3, 5, and 7 to find out the effects of stemming algorithms on the classification. The results of classification obtained from applying stemming and using the raw form of terms are compared, and the raw form of terms gives more accurate classification results. The effects of n-gram based feature extraction, and feature selection methods that are our proposed standard deviation based method, well-known information gain, and chi-square algorithms are compared. The experimental results indicate that the n-gram feature extraction and standard deviation-based feature selection algorithms give the best results and these methods improve the classification accuracy positively.

Keywords: Text classification, preprocessing methods, feature extraction, feature weighting, feature selection

ÖZ

YÜKSEK LİSANS TEZİ

NİTELİK ÇIKARIMI YÖNTEMLERİNİN TÜRKÇE METİNLERİN SINIFLANDIRILMASINA ETKİSİ

Özge AKDOĞAN

ÇUKUROVA ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
BİLGİSAYAR MÜHENDİSLİĞİ ANABİLİM DALI

Danışman : Prof. Dr. Selma Ayşe ÖZEL
Yıl: 2018, Sayfa: 81
Jüri : Prof. Dr. Selma Ayşe ÖZEL
: Doç. Dr. Zekeriya TÜFEKÇİ
: Dr. Öğr. Üyesi Alper Kamil DEMİR

Bu tezin amacı farklı türlerdeki Türkçe belgelerin sınıflandırılması için en etkin nitelik çıkarımı yöntemlerinin belirlenmesi ve etkili bir nitelik seçme yönteminin önerilmesidir. Çalışmada 1150 Haber, 3000 Tweet, Türkçe Email ve 25 Yazar veri kümeleri üzerinde deneyler yapılmıştır. En iyi terim ağırlıklandırma yöntemini belirlemek için 5 farklı yöntem (*tf, tp, logtf, normtf, tf*idf*) denenmiş ve bunların içerisinden en başarılı sonuçları veren *tf* ve *tf*idf* yöntemlerinin en başarılı olduğu belirlenmiştir. Sınıflandırma sonuçlarına olumlu bir etkisi olduğu için tüm metinlere durdurma kelimelerininin elenmesi önişlemi uygulanmıştır. Ağırlıklandırma yöntemini belirleme ve durdurma kelimelerinin etkisini ölçme deneyleri metinler içerisindeki terimlerin ham halleri kullanılarak yapılmıştır. Kelime kökü bulma algoritmalarının da sınıflandırılmaya etkisini araştırmak için metinlere Zemberek, Affix Stripping ve Fixed Prefix 3, 5, 7 olmak üzere 5 farklı kök bulma algoritması uygulanmıştır. Kelime kökü alınarak ve kelimelerin ham halleri ile yapılan sınıflandırma sonuçları karşılaştırılmış, kelimelerin ham halleri ile yapılan sınıflandırmanın daha başarılı sonuçlar verdiği gözlenmiştir. Nitelik çıkarımı yapmak için kelime n-gramı, nitelik seçimi için standart sapma tabanlı bir yöntem ile bilgi kazancı (information gain) ve ki kare algoritmaları uygulanmıştır. Terimlerin ham halleri kullanılarak kelime n-gram nitelik çıkarımı ve standart sapma nitelik seçimi en iyi sonucu vermiştir.

Anahtar kelimeler: Metin sınıflandırma, önişleme yöntemleri, nitelik çıkarımı, nitelik ağırlıklandırma, nitelik seçimi

EXTENDED ABSTRACT

Thanks to the development of information technology and the internet usage on mobile devices, the size of data has been incredibly large over the time. Extraction of meaningful information from the large sized data is called as data mining which is used in many different areas. One of the most important application area of data mining is text mining which aims to obtain interesting and meaningful information by using several methods from the data in the form of text documents. Text mining includes various application areas such as classification of texts, clustering of texts, emotion analysis, subject extraction from texts, text summarization, author identification, etc. Text mining has been done for many different languages such as Turkish, English, Arabic, Slovak, etc. Thanks to the text mining, studies for natural language processing have increased over the time.

In this study, our aim is to investigate the effects of preprocessing methods such as feature extraction, stemming, stopwords removal, term weighting, and feature selection on classification accuracy of Turkish texts from different problem domains. Therefore, we choose four datasets that are 1150 News, 3000 Tweet, Spam Email, and 25 Author datasets from four different text classification domains that are topic classification, sentiment analysis, spam e-mail detection, and author identification respectively, to determine how preprocessing methods affect different text classification problems in Turkish. We apply Naïve Bayes Multinomial as classifier since it is one of the best classifier for text documents in general.

As we study the effects of preprocessing methods on the classification accuracy of Turkish texts from different classification problem domains, first of all, we try to determine the best feature weighting scheme and compare five weighting methods that are *tf*, *tp*, *logtf*, *normtf*, and *tf*idf*. After that we study effects of stopwords removal. As a result of these two experiments, we observed that using *tf* or *tf*idf* weighting methods and eliminating stopwords give the highest

classification accuracies. Therefore, we use these two weighting methods and remove stopwords in the subsequent experiments.

In the third experiment, we measure the effects of stemming and try to determine the best stemmer for Turkish text classification. To reach our goal, we compare Zemberek, Affix Stripping, Fixed Prefix 3,5, and 7 stemming algorithms with the non-stemming case. Stemming is used to eliminate suffixes and take root forms of the terms so that it allows us to extract smaller number of features. In this experiment the best classification accuracies are obtained when stemming is not applied for the feature extraction. This result occurred due to the fact that Turkish is an agglutinative language where applying stemming may change the meaning of words, therefore reduces classification accuracy.

In the fourth experiment, we study the effects of using term n-grams as features. This experiment is done by taking 1-2 grams, and 1-2-3 grams of words. In the previous experiment, we have observed that the raw forms of the words give more successful results, therefore n-grams are computed by using the raw forms of the words. The inclusion of the word n-grams as features has positive effect on the accuracy of the classification in our experimental evaluation.

In this thesis we also propose a simple feature selection method which is based on the standard deviation of the occurrences of the words in the datasets. As the fifth experiment, the effect of the proposed feature selection method on the classification performance is studied. In this experiment, feature selection is applied to features that are non-stemmed terms obtained from the datasets. When we examine the classification results, it has been observed that the features that are selected by the proposed method yield more accurate classification with respect to classification made by using all features.

In the last experiment we compare the performance of our proposed method with the well-known feature selection methods namely information gain and chi-square, which are used in text classification studies. As a result of this experiment we also observed that, our proposed method has better classification

accuracy performance with respect to information gain and chi-square especially for multi-class classification problems.

As a result of this study we can conclude that, stemming is not a good preprocessing technique for Turkish text classification, as Turkish is an agglutinative language. Therefore, better stemming algorithms are needed. Word n-grams also improve classification accuracy, but they increase the feature space too much. As weighting methods *tf* and *tf*idf* both have the best performance and they can be used in Turkish text classification applications. Removing stopwords and applying feature selection improves the classification accuracy for all datasets.

ACKNOWLEDGEMENT

First of all, I would like to express my endless thanks and appreciation to head of Computer Engineering Department and my thesis supervisor Prof. Dr. Selma Ayşe Özel for all respectable knowledge, support and experience during my graduate education.

I would like to thank members of my MSc thesis jury, Assoc. Prof. Dr. Zekeriya TÜFEKÇİ and Assist. Prof. Dr. Alper Kamil DEMİR for their constructive and lead-in suggestions and corrections.

I would like to thank faculty members of Çukurova University Computer Engineering Department.

My special thanks for my family, my husband Mehmet YARAN and my lovely daughter Duru YARAN, for their support and love.

Finally, I would like to thank my dearest mother Pakize AKDOĞAN and my father Ömer AKDOĞAN who have always been with me all my life and trusting me.

CONTENTS **PAGE**

ABSTRACT I
ÖZ II
EXTENDED ABSTRACT III
ACKNOWLEDGEMENT VII
CONTENTS VIII
LIST OF TABLES X
LIST OF FIGURES XIV
ABBREVIATIONS XVI
1. INTRODUCTION 1
2. RELATED WORKS 7
3. MATERIALS AND METHODS 17
3.1. Materials 17
3.1.1. Datasets 17
3.1.1.1. 1150 News Dataset 17
3.1.1.2. 3000 Tweet Dataset 18
3.1.1.3. Turkish E-mail Dataset 18
3.1.1.4. 25 Author Dataset 18
3.1.2. Preto 19
3.1.3. WEKA Data Mining Tool 21
3.2. Methods 23
3.2.1. Stemming Algorithms 23
3.2.1.1. Zemberek 24
3.2.1.2. Affix Stripping 25
3.2.1.3. Fixed Prefix Stemming 26
3.2.2. Stopwords Removal 27
3.2.3. *n*-Gram 27
3.2.4. Term Weighting Methods 28

3.2.5. Feature Selection Methods 30
3.2.5.1. Using Standard Deviation as Feature Selector 30
3.2.5.2. Information Gain 31
3.2.5.3. Chi Square ($\chi 2$) 32
3.2.6. Naïve Bayes Multinomial Classifier 33
4. RESULTS AND DISCUSSIONS 37
4.1. Determination of the Best Term Weighting Method 39
4.2. Effects of Stopwords Removal 40
4.3. Determination of the Best Stemming Algorithm 42
4.4. Determination of Using N-Grams 44
4.5. Effects of Using Standard Deviation Based Feature Selection 46
4.6. Effects of Using Information Gain as Feature Selector 60
4.7. Effects of Using *Chi Square ($\chi 2$)* Measure as Feature Selector 61
4.8. Effects of the Feature Selection to the Classification Time 63
4.9. Comparison with Earlier Studies and Discussions 64
5. CONCLUSION AND FUTURE WORK 69
REFERENCES 71
CURRICULUM VITAE 75
APPENDIX 76

LIST OF TABLES **PAGE**

Table 2.1. Class distributions of datasets and number of instances 12

Table 4.1. Number of classes, number of instances, total number of words, number of unique words and number of characters for each dataset. 37

Table 4.2. Average number of characters per word (i.e., average word length), and average number of characters per document for each dataset. 38

Table 4.3. F-Measure values of the NBM classifier for different term weighting methods 39

Table 4.4. Number of terms for each dataset.....40

Table 4.5. F-measure values for the datasets with NSNS and NSWS preprocessing.....41

Table 4.6. Number of unique tokens after applying stemming algorithms 42

Table 4.7. The classification results in terms of F-Measure values for Zemberek, Affix Stripping, Fixed Prefix 3,5,7 stemming algorithms.....43

Table 4.8. The number of tokens generated after applying term 1-2grams, and 1-2-3grams feature extraction methods 44

Table 4.9. The number of tokens left after filtering is applied to 1-2grams and 1-2-3grams forms 45

Table 4.10. F-Measure values for term n-grams 45

Table 4.11. The number of terms whose standard deviation is greater than zero.....47

Table 4.12. The number of terms selected after taking top ranked 0.1 percent features having standard deviation greater than zero.....47

Table 4.13. F-Measure values for classification with the selected features by using top ranked 0.1 percent features.....48

Table 4.14. The number of terms selected after taking top ranked 0.5 percent features having standard deviation greater than zero.....49
Table 4.15. F-Measure values for classification with the selected features by using top ranked 0.5 percent features.....49
Table 4.16. The number of terms selected after taking top ranked 1 percent features having standard deviation greater than zero.....50
Table 4.17. F-Measure values for classification with the selected features by using top ranked 1 percent features.....51
Table 4.18. The number of terms selected after taking top ranked 5 percent features having standard deviation greater than zero.....52
Table 4.19. F-Measure values for classification with the selected features by using top ranked 5 percent features.....52
Table 4.20. The number of terms selected after taking top ranked 10 percent features having standard deviation greater than zero.....53
Table 4.21. F-Measure values for classification with the selected features by using top ranked 10 percent features.....53
Table 4.22. The number of terms selected after taking top ranked 15 percent features having standard deviation greater than zero.....54
Table 4.23. F-Measure values for classification with the selected features by using top ranked 15 percent features.....54
Table 4.24. The number of terms selected after taking top ranked 20 percent features having standard deviation greater than zero.....55
Table 4.25. F-Measure values for classification with the selected features by using top ranked 20 percent features.....55
Table 4.26. F-Measure values for classification when all features having standard deviation greater than zero are used.....56
Table 4.27. Number of terms selected for each dataset when standard deviation based feature selection is applied 57

Table 4.28. F-Measure values obtained when standard deviation-based feature selection is applied and *tf* weighting is used 58
Table 4.29. F-Measure values obtained when standard deviation-based feature selection is applied and *tf *idf* weighting is used 59
Table 4.30. Number of terms selected after applying IG based feature selection.....60
Table 4.31. Comparison of SD and IG methods with no feature selection case 60
Table 4.32. Number of terms selected after applying *Chi Square* based feature selection.....62
Table 4.33. Comparison of SD and Chi Square methods with no feature selection case.....62
Table 4.34. Time Required (in seconds) for Classification when the Best F-Measure Values Obtained.....63

LIST OF FIGURES **PAGE**

Figure 3.1. Graphical User Interface of PRETO 19
Figure 3.2. General view of the processing steps of PRETO 20
Figure 3.3. WEKA initial graphical user interface 21
Figure 3.4. Explorer interface of WEKA 22
Figure 3.5 A view from the Arff file 23
Figure 3.6. An example for Zemberek 25

ABBREVIATIONS

Arff	: Attribute Relation File Format
FSM	: Finite State Machine
IDF	: Inverse Document Frequency
IG	: Information Gain
NBM	: Naive Bayes Multinomial
NLP	: Natural Language Processing
NSNS	: No Stemming No Stopword Removal
NSWS	: No Stemming With Stopword Removal
tf	: Term Frequency
tf*idf	: Term Frequency Inverse Document Frequency
WSWZ	: With Stopword Removal With Zemberek
WSWA	: With Stopword Removal With Affix Stripping
WSWF3	: With Stopword Removal With Fixed Prefix3
WSWF5	: With Stopword Removal With Fixed Prefix5
WSWF7	: With Stopword Removal With Fixed Prefix7

1. INTRODUCTION

The development of Information Technology has resulted in the rapid growth of e-mail usage, forming blogs, content sharing in social media, e-commerce activities on the internet, which cause to increase in the dimensions of the online data. This high dimensional and large-scale data leads to extreme information generated in electronic environments and also information pollution. Studies on databases and information technology have shown the need to use meaningful information within these data and the use of these large-scale data. As it is impossible to process large data manually, automatic techniques to gather and analyze this data has been necessary over time. Extracting information from these large-scale data by using some machine learning techniques is called as data mining. Data mining makes some predictions by using the large amounts of data. Data mining is described as “analysis of large observational datasets to summarize data those are understandable and useful to users with new methods and to find unexpected relationships between them” (Hand et. al., 2001). In other words, this large data needs to be analyzed in a meaningful way as well as stored efficiently. Data mining has emerged to reach the right results from the necessary and unnecessary data in this entire electronic environment.

Data mining is the search for the relations and rules that will enable us to make predictions about the future from a large amount of data, by using computer programs (Alpaydın, 2000). For this reason, in every field where large data exists, data mining can be used effectively. Today, data mining applications are widely used in many fields where decision-making process is a need. Successful practices of data mining are seen in many branches, for example; marketing, biology, banking, insurance, stock exchange, retailing, telecommunications, genetics, health, science and engineering, criminology, industry, intelligence etc. (İnan, 2003; Albayrak, 2008; Akgöbek & Çakır, 2009). Most of these agencies and organizations use data mining together with statistics, pattern recognition, and

other important tools. Companies often use data mining in order to provide higher quality services to their customers, to recognize and classify them according to their behaviors, to obtain more detailed information about them and to increase customer satisfaction and reduce customer complaints by offering them smart marketing options. In another example, banks use data mining in bank credit card fraud detection, and in credit card demand of customers to make evaluation of this claim.

With the widespread use of the Internet, web, and mobile devices; the unstructured and unprocessed text data flow in different file formats like pdf, txt, doc, html etc. from different sources, has accelerated thus, it has been difficult to control these data. Since large amounts of text data have increased in size, it has become necessary to process, classify, cluster, and understand these documents over time. At this point, data mining has been introduced. One of the most important fields of data mining is text mining. Text mining is considered to be the subdivision of data mining. The most important difference that distinguishes text mining from data mining is that pattern is mined from natural language texts rather than proper databases in text mining (İlhan et al., 2008).

Additionally, in text mining, natural language processing activities are also being carried out. Natural Language Processing (NLP) is subordinate to artificial intelligence and linguistics. NLP is an engineering field that takes the design of computer systems that have a main function of natural language analysis, understanding, interpretation and generation (Knight, 1991). Thanks to natural language processing exercises, the increase of human-computer interaction has been achieved (İlhan et al., 2008).

Text mining techniques are divided into 4 basic categories: classification, clustering, association analysis, and knowledge extraction. Among these techniques, classification technique is used in this study. Purpose of document classification can be defined as obtaining a certain number of predefined categories

by looking at the properties of a document to determine which one would be included in.

The increasing number of documents on the Web revealed the need to classify documents into pre-determined classes. It has great importance in the regulation of large amounts of data automatically. Text classification looks at properties of a document and a number of predefined categories to determine the ones to be included. Text classification has very important roles in information management and access to mission. Text documents are unstructured data and written in the main natural languages. To apply data mining techniques to text data, the unstructured data must be preprocessed at first. There are several preprocessing methods such as term extraction, term weighting and feature selection. Studies on text mining in general, are on English documents. Limited numbers of studies have been conducted for Turkish. Therefore, in this thesis we study text classification for Turkish text documents.

There is no standard rule of writing texts, therefore these unstructured documents cannot be understood by computers automatically. Firstly the data should be preprocessed. At the end of this preprocessing process, the data which is analyzed is cleaned from noise and unnecessary data, and is converted to a specific format. For this reason, data preprocessing is the first step for classification algorithms.

Stemming is one of the commonly used preprocessing methods in the document classification. It is used especially when the documents have big size and need to be classified accurately. Stemming provides us to find the root form of any given term. Therefore, the number and length of the roots should be as little as possible to speed up the classifier.

Stemming algorithms are widely used to search for morphological diversification in databases. There are stemming algorithms in the literature for various languages (e.g. English, Slovak, German, Greek). There are no exact rules for stemmers of all languages because stemmer’s morphological analysis structure

must be appropriate to the applied language. In some of the languages, words can take suffixes; whereas in some of the languages words can take prefixes. Therefore, stemmers are language dependent. There are several stemmers for different languages, however there exist only a few stemmers created for Turkish such as Zemberek, Affix Stripping, and A-F Algorithm.

Turkish language has an agglutinative structure. In Turkish, millions of words can be derived from a few roots by appending suffixes to the roots. Turkish language consists of 21 consonants, 8 vowels and the language has 29 letters in total. As Turkish is an agglutinative language, a new word can be created with one or more suffixes from a simple root. Therefore, the number of unique terms that can be extracted from a document collection is large. Using a stemming algorithm provides to reveal the words from the same root and the number of distinct terms extracted from a document collection is reduced.

In classification studies, each word is used as a feature; therefore stemming algorithms, as well as the n-gram method have been used to reduce feature space. N-gram is the string of n characters in a character zone. N-gram based method of classifying the document is based on a process which counts the frequency of used n-grams in the documents. N-gram method is a simple and reliable method, and works regardless of the language. So, detailed grammar rules or a dictionary of a particular language are not required to apply n-gram method.

The words which are used in the language often and can be found in almost every text documents are called “stopwords”. These terms are generally independent from the content and they can be conjunctions, prepositions, numbers, abbreviations etc. These terms may differ from language to language. For Turkish language, some stopwords are “one, every, very, what etc.”. In some cases stopwords are removed from texts. The stopword removal operation is applied in the pre-processing phase of the text mining to reduce feature space and noise. These terms are removed from the texts because they do not affect the classification in the positive direction.

A computer cannot understand or interpret the words used in the document. The words in the text need to be transformed into a form that the computer can understand, and term weighting methods are used to accomplish this. In term weighting, a term in the document is assigned a numerical weight, and by using the weights of all the terms in the document a numerical vector for the document is formed. Most common term weighting methods are term frequency (*tf*), and term frequency * inverse document frequency (*tf*idf*). Term frequency is the observed frequency of terms in the document; *idf* is a statistical measure used to calculate how important a term is for that text, and *tf*idf* is the multiplication of these two measures.

Feature selection is used in many different areas as medicine, image processing, data mining, and text mining to reduce problem size to be solved. There are some algorithms to do feature selection on text mining. Main purpose of feature selection in text mining is to remove terms which are useless for classification to obtain more successful results. Also, removing irrelevant and useless features decreases the size of datasets. The process of choosing a subset of the terms within the training set and employing this subset in the classification of text as features is called feature selection. Two primary objectives are determined for feature selection: The first of these objectives is to make training and applying a classifier more effective by way of decreasing the size of the effective vocabulary. Secondly, it is aimed to increase classification accuracy through removing noise features which lead to an increase in the classification error on the new data when added to the document presentation.

According to the related literature, there are many studies concerned with text mining on different languages. In majority of the text mining studies, standard preprocessing steps are applied, and the effects of classification methods are studied. However, in this thesis, our aim is to show the effects of all preprocessing steps that are stemming, n-grams, stopword removal, term weighting, and feature selection on text mining from different domains for Turkish language. There are

several studies for text mining form Turkish documents, however majority of these studies focus on a particular domain such as news classification, author identification, spam e-mail detection, etc., and for all different domains the same preprocessing steps are applied. In this thesis, our aim is to investigate the effects of all preprocessing steps over four different text classification problems for Turkish language and to show the differences and similarities between the domains. Therefore, we try to determine the best preprocessing steps are to be done for each text mining problem separately. In this thesis we investigate the effects of

i. using three different stemmers that are Zemberek, Affix Stripping, and Fixed Prefix stemming as well as not applying stemming, and try to determine the best stemming method in general,
ii. using term n-grams as features,
iii. using stopword removal or not,
iv. using five different term weighting methods that are *tf*, *tp*, *normtf*, *logtf*, and *tf*idf*, and try to determine the best weighting method for Turkish texts, and
v. applying feature selection.

Additionally, we propose a new feature selector, and compare its performance with well-known methods.

The rest of this thesis is organized as follows; In chapter 2 related works are summarized for Turkish language. In chapter 3, materials and methods which are used in this thesis are explained. In chapter 4, results obtained from our study are presented and discussed. In chapter 5, our results are compared with the related works, and the contributions made in this thesis are discussed. Suggestions for future studies are presented in chapter 5.

2. RELATED WORKS

In this section, we summarize the Turkish text classification studies which investigate the effects of preprocessing methods on different text classification domains.

The earliest study on the effects of preprocessing of Turkish text documents belongs to Ekmekçioğlu and Willet (2000), who have studied the effects of stemming on Turkish text retrieval. This study belongs to information retrieval domain, not to text mining domain. However, it is an important study to show the effects of stemming for Turkish texts. In this study, Turkish papers between the years of 1991-93 that contains the economic and political issues are used as the dataset. This study is a text retrieval system which employs OKAPI system. OKAPI calculates the weight of each term in the documents and queries by using the probabilistic model. In this study, PC-KIMMO is used for stemming. As a result, it has been observed that stemming applied querying gives more related results.

Later, Sever and Bitirim (2003), worked with a new algorithm FINDSTEM for finding the root of terms in documents and queries. This study is also in information retrieval domain and aims to develop a new stemmer for Turkish. The dataset used contains the 2468 Turkish-based legal documents. FINDSTEM is compared with Turkish LM (Longest-Match) and the AF algorithms. Regionalized version of the SMART system has been used for the experiments. The experimental results show that, FINDSTEM algorithm has better performance than other stemming algorithms and it is observed as more effective.

The above studies show that, applying stemming to Turkish text documents and queries improves the performance of information retrieval systems. However, studies on Turkish text mining have begun after the year 2000. One of the earliest studies on Turkish text mining belongs to Amasyalı and Diri (2006), who have researched at the effect of using n-grams for classifying Turkish texts. They studied

three different classification problems that are classification of: author, type of the text, and gender of the author. Datasets are collected from Turkish newspapers. For the text classification, a dataset containing 18 different authors' articles is used, in other words it includes 18 classes. For the classification of types, 35 articles of 18 different authors are chosen. There are three different classes in this dataset: political, popular interest, and sports. The same dataset is used for gender experiments by partitioning 18 authors as 14 males and 4 females. Naive Bayes (NB) classification method, Support Vector Machine (SVM), C4.5, and Random Forests (RF) methods are used for the classification; and Correlation-based Feature Selection (CFS) is used for the feature selection. For each classification problem, 2 different n-gram models are used namely; bi-gram and tri-gram. When comparing the results, feature selection increases the classification success of the three classification problems. While the bi-gram is the best model for the author, the two n-gram models for types and gender give the same classification results. While NB is the best classifier for the author, the best classifier is SVM for types and gender.

Yıldız et al. (2007), proposed a new feature vector computation method in which the class weights are used instead of the weight of the words in the texts, and the sum of these class weights is normalized. The dataset used in this study is collected from daily newspapers as Hürriyet (www.hurriyet.com.tr), Vatan (www.vatanim.com.tr) and Sabah (www.sabah.com.tr). Articles from economy, magazines, health, politics, and sports classes are collected to form the dataset. Zemberek is used for finding roots. Success of the machine learning techniques that are NB, SVM, C4.5, K-Nearest Neighborhood (KNN), and RF are compared and the highest achievement is observed by using the NB as 96% classification accuracy.

Çataltepe et al. (2008) have analyzed the performance of classifiers when the longest or shortest roots found by a stemmer are used. Also they have analyzed the effect of using only the consonants in the roots. Two datasets namely, Milliyet and Vikipedi having 5 different categories are used for experimentation. Milliyet

dataset has world, economy, actual, politics, and sport categories such that each category has 200 documents. Vikipedi categories are art, sport, politics, history, and technology; and each category has 200 documents, too. Stopwords are removed from documents. Zemberek is used for stemming. After the words are stemmed, the longest root, the shortest root, and the longest 4, 3, 2 groups of letters are divided into 2 groups and they are grouped as vowel + consonant and only consonant. Experiments are conducted by taking the 5% and 20% of the most frequently used words from the selected roots. Centroid and Support Vector Machine classification algorithms and *tf*idf* term weighting method are applied. In this study, the use of different stem lengths and stems without vowels are examined. As a result, 20% of the most frequently used terms in each document are taken, the results of the experiment by removing the vowels from roots and the results received by using all the letters are close to each other. When a large number of documents needs to be classified in a short time, only the consonants are taken from the words to form features, therefore the document processing time can be reduced.

Güran et al. (2009) have analyzed the effects of using n-grams on Turkish text classification. They have collected a dataset from the Web and it consists of 600 documents from 6 categories namely; auto, politics, medicine, magazine, economics, and sport. Each category has 100 documents. Each document is represented by using unigrams words, bigrams words, and trigrams words separated with a pipe. *tf*idf* weighting is used and for document preprocessing, PC-KIMMO, two level morphological parser is applied. Naive Bayes, Complement Naive Bayes (CNB), Multinomial Naive Bayes (MNB), C4.5 Decision Tree (J48), K-Nearest Neighbor classification methods are used for classifying the documents. From the classification results, the most successful result for all classifiers is the uni-gram. NB classifier gives the highest success rate for tri-grams, while the MNB classifier has the highest percentage of success for uni-grams and bi-grams.

Torunoğlu et al. (2011) analyzed the effect of preprocessing methods in text classification on Turkish news texts. They have used 3 datasets namely; 1150 News, Milliyet_9c_1k, and Hurriyet_6c_1k. 1150 News dataset consists of 1150 Turkish news texts in 5 classes (i.e., economy, magazine, health political, sports) and 230 documents for each category. Milliyet_9c_1k includes text from columns of Turkish newspaper Milliyet from years 2002 to 2011. There are nine categories and 1000 documents for each category. The categories of this dataset are café, world, region, economics, current, politics, sports, Turkey, and life. Hurriyet_6c_1k dataset includes the documents in Turkish newspaper Hürriyet from 2010 to 2011. It contains six categories and 1000 documents for each category. Categories are world, economy, current, politics, sports, life. Both Milliyet_9c_1k and Hurriyet_6c_1k datasets are collected by running a crawler over the websites of the newspapers. Two term weighting methods as binary weighting (*tp*) and term frequency (*tf*) weighting are used. For the classification, Naïve Bayes, Naïve Bayes Multinomial, Support Vector Machine, and K-Nearest Neighbor methods are applied. Zemberek and Fixed Prefix 3, 5, 7 stemming algorithms are used in experiments. Stopword filtering is also applied, the effect of stopword removal and stemming is only visible on small training dataset size. On these small training set sizes Zemberek is the best stemmer for SVM algorithm. Naïve Bayes Multinomial classifier is better with stopword removal and fixed prefix 5 stemmer.

Another study made on Turkish text document mining belongs to Tunalı and Bilgin (2012a), who have investigated the effects of different stemming methods on Turkish text clustering using news datasets that are Milliyet and NTV datasets. Milliyet dataset has 1455 articles published in 2006, collected from website of the newspaper Milliyet. The economics, politics, and sports categories are used, and each category has 485 documents. NTV dataset consists of the documents published by NTV television on their website between 2009-2010 years. NTV dataset has 19,476 documents and it has 9 different categories: as

science, world, economy, art, health, Turkey, life, and green. The smallest category has 535 documents; the largest category has 5806 documents, the average of the number of documents per class is 2,164. For stemming; Zemberek, Affix Stripping, and Fixed Prefix 3, 5, 7 stemming algorithms are compared, and *tf*idf* term weighting method is used to weight the terms in the documents. Stopword filtering is applied to the documents and to cluster the documents, the Spherical K-Means algorithm is used. Clustering performance is measured according to purity, entropy, normalized mutual information and F-measure values. Consequently, it has been observed that stemming is not very effective in clustering the Turkish documents, but when the stemming is applied to the texts, the size of the document-term matrix has decreased. Because of this reason, they think that stemming algorithms should be absolutely used in the clustering of Turkish documents.

Uysal and Serkan (2013) have studied the influence of the preprocessing tasks on the text classification for two different domains and languages that are Turkish and English. They have used 4 preprocessing methods namely; tokenization, stopword removal, lowercase conversion, and stemming. Zemberek and Fixed Prefix stemming algorithms are used for Turkish, and Porter's stemming algorithm is applied for English. 2 different text classification domains that are spam e-mail detection, and news classification in two different languages namely Turkish and English are studied. The e-mail dataset contains 300 training and 100 testing samples for each class as spam and legitimate. Turkish news dataset is a subset of Milliyet collection, and English news dataset is a subset of Reuters-21578. There are 10 classes in the both news datasets. For feature selection chi-square (CHI2) method is used. To classify datasets, Support Vector Machine classifier is applied and Micro-F1 score is computed. Thorough experimental analysis elucidated that significant improvement may be ensured through appropriate combinations of preprocessing tasks on the basis of domain and language while the accuracy may also be reduced by inappropriate combinations.

Certain preprocessing steps such as feature extraction and selection in text classification are as significant as the classification step. Despite specific preprocessing tasks employed to ensure an improvement in the classification success with regards to accuracy and dimension reduction irrespective of domain and language, we can talk about no distinctive combination of preprocessing tasks that generate effective classification results for every domain and language examined.

A comprehensive research about text classification is made by Amasyalı et al. (2012) who have used *tf*, *tf*idf*, *binary*, *log*, *normalize*1 and *normalize*2 term weighting methods for 6 different datasets that are "ruh hali" (for emotion classification), "film yorumları" (for sentiment analysis), "köşe yazarı" (for author identification), "cinsiyet" (for gender identification), "haber" (for news classification), and "şiir" (for author identification). Information about the datasets are given in Table 2.1.

Table 2.1. Class distributions of datasets and number of instances

Dataset name	# of classes	# of instances	# of instances per class
Ruh hali	4	157	38-40
Film Yorumları	3	105	35
Köşe Yazarı	18	630	35
Cinsiyet	2	105	50-55
Haberler	5	1150	230
Şiir	7	1140	20

14 different text representation methods that are listed as follows are used in the study.

i. In the study, 6 different methods *tf*, *tf*idf*, *binary*, *log*, *normalize*1, and *normalize*2 are used in weight calculations for word stems, word types, n-grams, functional words, suffixes and concept generalization feature groups.
ii. Only the 19 features consist of the punctuation marks, word counts, sentence counts, inverted sentence counts, letter counts, affix counts, average words and letters in sentences, average letters, and affix counts in words are used.
iii. Stemmed words are taken to form the features. In the study, Zemberek is used to find the roots of the words.
iv. Only the word types such as noun, adjective, adverb, and preposition are used. Zemberek is utilized to find the word types.
v. Feature group formed by word and letter n-grams.
vi. Words those are important in determining the author's styles, although they have no meaning on their own (such as preposition and conjunction).
vii. A group of features that are expressed according to the types of affixes they had is employed. Zemberek is used to determine the affixes.
viii. A group of features that are formed by combining closely related terms in the text.
ix. Collocation matrix that calculates the counts of the words used together in the text is used to represent the words in grouping and in semantic space.
x. Terms are filtered based on the specified minimum and maximum number of existences in the text, and the remaining terms are used as features.
xi. Reducing the sizes of texts by performing a Singular Value Decomposition operation on the text-word matrix without using class information.
xii. By using the coordinates of the words that represent the texts in a semantic space, and taking the average of the coordinates of the words that exist in the document.

xiii. By using concepts as features instead of words in the documents. In this representation, Turkish Wordnet is used to determine concepts for the words.

xiv. By using the method proposed by Amasyalı et al. (2012), where firstly the frequencies of the words are found in each class and the words are transformed into a point in space with a dimension of a number of classes. Then, the words are grouped into clusters where number of clusters is equal to number classes. At the end of this process, each word belongs to one of the clusters. The texts are expressed with frequencies of the clusters. The frequency of a cluster in a text is the sum of the frequencies of all words in that cluster that occurs in the text. Thus, texts are expressed with a vector which has size class number instead of word number (Amasyalı et al., 2012).

They have used K-Nearest Neighbour, Decision Tree-C4.5, Support Vector Machine, Naïve Bayes, and Random Forest classification methods. According to the experimental results, n-grams based text representation have been more successful than other methods. N-gram with ***binary***, ***log***, and ***normalize***1 feature weighting gave better results than other methods. The proposed method is the most successful method for movie review dataset.

Açıkalın and Beyazıt (2016) have conducted a study to investigate the importance of preprocessing in the classification of Turkish texts. They have used paper abstracts in journals and conferences as a dataset. They have used 2 datasets in the study; there are 18 classes in the first dataset and there are 34 classes in the second dataset. In the study Latent Dirichlet Allocation is applied to do text preprocessing. Zemberek and fixed prefix with length 5 is used as the stemming methods. Naïve Bayes, Support Vector Machines, and Random Forest were applied as classifiers. As a result, they have observed that classifier performance increases in both stemming methods.

A study on term weighting and feature extraction for emotion analysis was conducted by Parlar and Özel (2018). They used the QER method and the Chi Square method for feature selection in the study. *tp, tf, tf*idf* term weighting methods are applied and experiments are performed on Movie, Book, DVD, Electronics, and Kitchen datasets. There are 1057 positive and 978 negative documents in the Movie dataset; and 700 positive and 700 negative documents in the other datasets. Naïve Bayes Multinomial is used as classifier. As a result of these experiments, they observed that the *tp* and *tf* term weighting methods are more successful than the *tf*idf*. The most successful result for Movie dataset is 83.4% with *tp*, the best accuracy for the Book dataset is 83.2% with *tf*, the most successful result for DVD dataset is 79.3% with *tf*, the most successful result for Electronics dataset is 81.5% with *tf* and the most successful result for Kitchen dataset is 78.1% with *tf*. When the QER and Chi Square feature selection methods are applied, they observed that the accuracies for *tf*idf* increase significantly. In addition, they obtained better results with QER feature selection than chi square. In emotion analysis studies, the *tf*idf* term weighting method gives better results when feature selection is applied.

When we compare the previous studies with this thesis as summarized in Appendix 1, the most important difference of our study from the previous studies is that we compare effects of preprocessing methods on four different text classification problems, however the previous studies make this comparison only for one problem domain, or compare only a few methods on small number of different problem domains. As shown in Appendix 1, several stemmers that are Zemberek, Fixed Prefix, Affix Striping, PC-KIMMO, Findstem, A-F, and L-F algorithms have been used in the previous studies. However, majority of the researchers usually preferred Zemberek and Fixed Prefix in the previous studies. In our study, we use Zemberek, Affix Stripping and Fixed Prefix 3, 5, 7 for stemming, and try to compare their performances on four different text classification problems.

As stopwords are often repeated frequently in text, they do not affect classification accuracy and for this reason stopwords are removed from the text in most of the previous studies. In this thesis, we also show the effect of removing stopwords or not for text classification from different problem domains.

In majority of the previous studies, *tf* and *tf*idf* methods have been used as term weighting method; and only in a few study *logtf*, *normalize1*, *normalize2* and *tp* weighting have been used together and compared. In this thesis study, we also use *tf, tp, logtf, normtf* and *tf*idf* methods and compare them for different text classification problems.

In the previous studies, Naïve Bayes, Naïve Bayes Multinomial, Support Vector Machines, C4.5, K-nearest Neighbor, and Random Forest have been used as classifiers. In this study, we use Naïve Bayes Multinomial as the classifier because it is one of the most successful classifiers for text classification problems in the literature. As our aim is only to make comparison of preprocessing methods, we think that using one successful classifier is enough.

We also propose a simple feature selection method which is based on standard deviation of frequencies of features. According to the literature, the researchers often use information gain, and chi square measures for feature selection. Therefore in this study, we compare our proposed standard deviation based method with information gain and chi square to show the effect of feature selection on different text classification problems.

In this thesis, we try to determine the best preprocessing methods for four different text classification tasks, and investigate the answer of the research question “Should we apply the same preprocessing methods for all text classification tasks? Or should we apply problem specific preprocessing methods?”.

3. MATERIALS AND METHODS

This section contains information about the datasets namely 1150 News, 3000 Tweet, Turkish Email, and 25 Author datasets; WEKA data mining tool used for classification, term weighting methods which are *tf*, *tf*idf*, *tp*, *normalized tf*, *log tf*; Naïve Bayes Multinomial classification model; preprocessing tool PRETO; stopwords; Zemberek, Affix Stripping and Fixed Prefix Stemming as stemming algorithms; and feature selection methods applied.

3.1. Materials

3.1.1. Datasets

The datasets used in this thesis consist of 1150 News, 3000 Tweet, Turkish Email, and 25 Author which have different lengths and characteristics, belonging to the four different text classification problems. For example, a document in 3000 Tweet dataset may have only 4 terms whereas a document in 25 Author dataset can have more than 300 terms. We chose these datasets as we want to compare methods for different text classification problems which have different number of classes, document types, and document lengths.

3.1.1.1. 1150 News Dataset

1150 News dataset (Amasyalı and Beken, 2009) has been collected by Nilgün DURSUNOĞLU and M. Fatih AMASYALI at Yıldız Technical University. The dataset contains newspaper articles written in Turkish from 5 different classes namely; economy, magazine, health, politics, and sport. News articles have been collected from various newspapers. Each class has 230 documents in it. The dataset is used to classify news documents with respect to their topics.

3.1.1.2. 3000 Tweet Dataset

3000 Tweet dataset (Amasyalı and Çetin, 2013) consists of 3000 documents each of which is a tweet written in Turkish from 3 different classes namely; positive, negative, and neutral. The dataset is collected by a group of students taking the course "Expert Systems" in Yıldız Technical University. In this dataset, positive class has 756 documents, negative class has 1287 documents, and neutral class has 957 documents. Positive, negative, and neutral classes have documents that have positive, negative, and netural feelings tweets, respectively, written in Turkish language. The dataset is collected for making sentiment analysis from Turkish tweets.

3.1.1.3. Turkish E-mail Dataset

This dataset is the first Turkish Email dataset which is collected by Semih ERGİN, Hüseyin YİĞİT, and Rıfat AYDIN as an Engineering Synthesis and Design Project disserted at Eskisehir Osmangazi University (Ergin et all., 2012). In this dataset there are 2 different classes which are called as normal email, and spam email. Each class has 400 documents in it. The dataset is used to make spam email detection for Turkish language.

3.1.1.4. 25 Author Dataset

25 Author dataset is derived from 2500 column dataset and created by Serkan ÇABUK and Burcu ÇELİK (http://www.kemik.yildiz.edu.tr) at Yıldız Technical University. 2500 column dataset is a very big dataset. It consists of 50 classes and 50 documents for each class, totally have 2500 documents. Dataset contains articles of various columnists. In this study, we have chosen 25 classes from 2500 column dataset randomly. The class names are the name of the authors who have written the articles. The aim of using this dataset is to make authorship attribution. Names of the classes are: Ahmet ÇAKAR, Ataol BEHRAMOĞLU, Atilla DORSAY, Aykan SEVER, Aziz ÜSTEL, Can ATAKLI, Cengiz ÇANDAR,

Ekrem DUMANLI, Emre KONGAR, Hasan PULUR, İclal AYDIN, İdil ÇELİKER, Kürşat BUMİN, Leyla İPEKÇİ, M. Ali BİRAND, Meltem GÜRLE, Mümtaz SOYSAL, Neşe YAŞİN, Nuri ELİBOL, O. Başak TANELİ, Okay KARACAN, Selim İLERİ, Tarhan ERDEM, Ufuk BOZKIR, and Yaşar SEYMAN.

3.1.2. Preto

PRETO is a platform-independent preprocessing toolkit which is developed for Turkish and English (Tunalı and Bilgin, 2012) text documents. Main components of PRETO is shown in Figure 3.1.

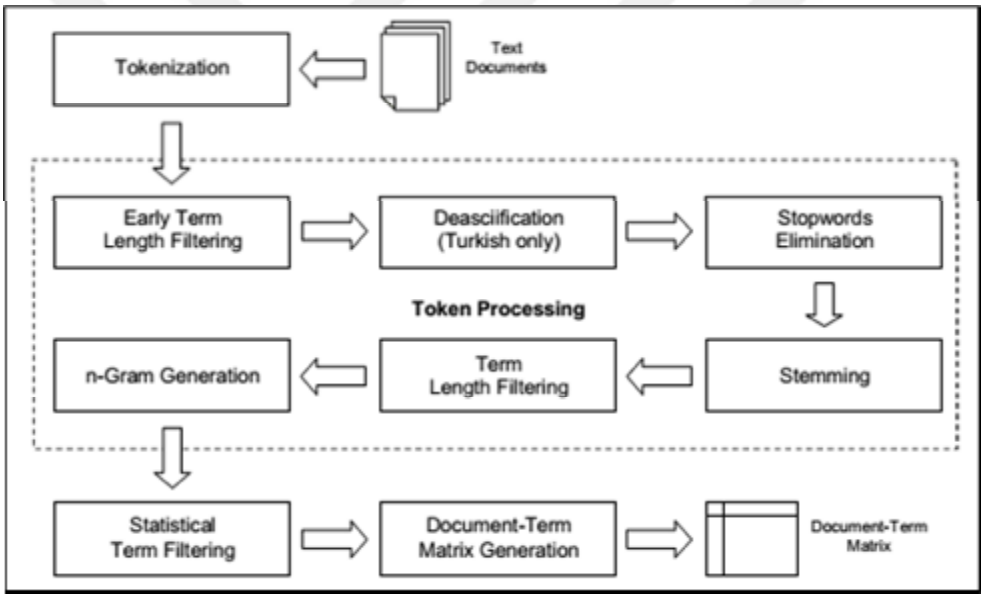

Figure 3.1. Architecture of PRETO

There are different preprocessing methods implemented in PRETO which offers wide range of options for stemming, stopword filtering, statistical term filtering, and n-gram generation (Tunalı and Bilgin, 2012b). Preprocessing steps available in PRETO are shown in Figure 3.2.

PRETO uses Zemberek, Affix Stripping, and Fixed Prefix Stemmers for Turkish; and Porter's stemming algorithm option for English uses.

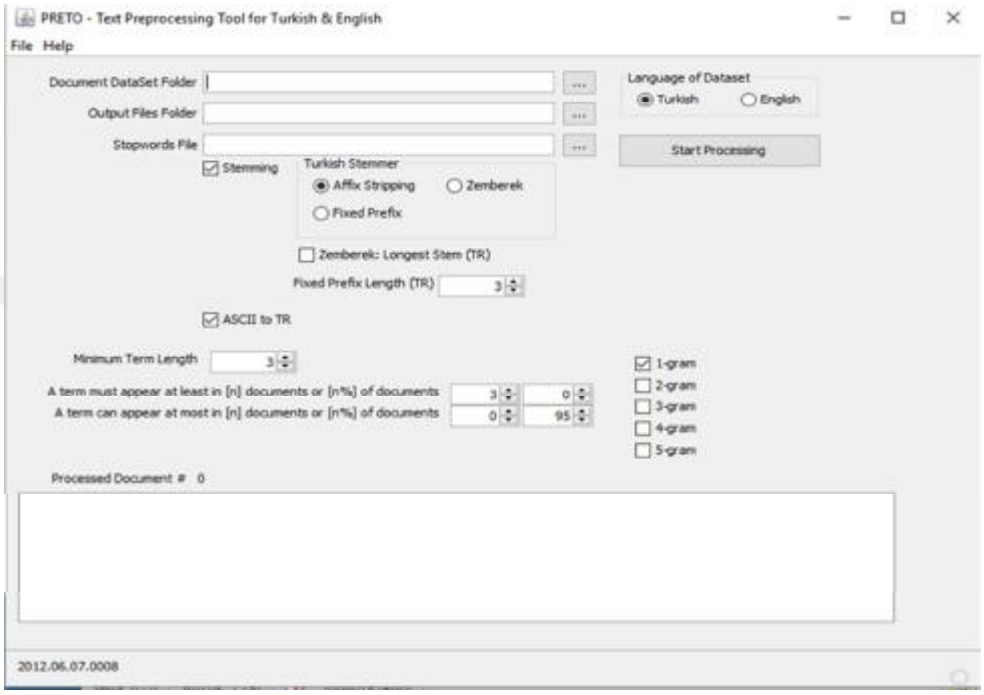

Figure 3.2. Graphical user interface for the processing steps of PRETO (Tunalı and Bilgin, 2012)

PRETO has several term filtering options. First one is the stopword filtering that removes stopwords from text file. Another filtering option filters the terms according to their lengths. For example, terms shorter than 4 characters can be directly filtered out. Furthermore, PRETO offers options for statistical term filtering for example terms having document frequency smaller than the user defined threshold are removed from the features list (Tunalı and Bilgin, 2012).

Once all the preprocessing steps have been concluded, PRETO produces document-term matrices with three distinct term weighting alternatives in two common sparse matrix file formats. Term frequency only (tf), term frequency * inverse document frequency (*tf*idf*) and normalized *tf*idf* (*tf*idf-norm*) are the

only available weighting alternatives. On the other hand, Coordinate List (COO) file format and List of Lists (LIL) file format are the two available sparse matrix file formats. (Tunalı and Bilgin, 2012).

3.1.3. WEKA Data Mining Tool

WEKA (Waikato Environment for Knowledge Analysis) (http://www.cs.waikato.ac.nz/ml/weka) is a popular suite of machine learning software which is developed in the University of Waikato in Australia under the GNU license and it is a java based data mining tool. WEKA's initial graphical user interface is shown in Figure 3.3.

Figure 3.3. WEKA initial graphical user interface

WEKA, which has 4 different working modes (Explorer, Experimenter, Knowledge Flow, and Simple CLI) (Witten and Frank, 2005), as shown in Figure 3.3, allows a number of data mining tasks such as data preprocessing, clustering, classification, regression, association rule mining, visualization, and feature selection. Of these working modes, Simple CLI is a basic command line interface that provides direct executions of WEKA through commands while Experimenter

carries out experiments and performs statistical tests between learning schemes. On the other hand, Knowledge Flow allows the same functions provided by the Explorer, but it does so with a drag-and-drop interface. One benefit of Knowledge Flow is that it contributes to progressive learning. Explorer is an environment that enables an exploration of data with WEKA. (Witten and Frank, 2005). Explorer is the widely used interface in WEKA and it is shown in Figure 3.4.

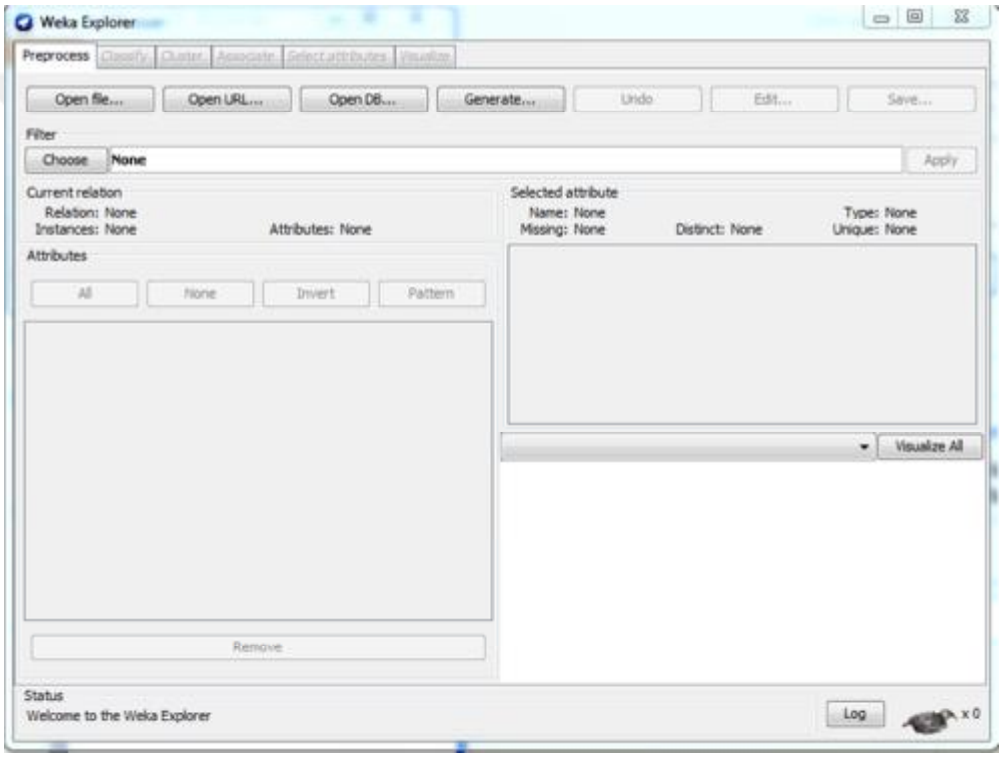

Figure 3.4. Explorer interface of WEKA

Explorer interface has 7 different tabs as: Preprocess, Classify, Cluster, Associate, Select Attributes, and Visualization. In preprocess tab, data to be analyzed is chosen and data is ready for processing. Classify tab has 71 algorithms and WEKA grouped them in 8 categories as: bayes, function, lazy, meta, misc, rules, and trees. Cluster tab can be used for clustering datasets. Association rules

are learned in Associate tab. Select attributes tab includes attribute selection methods. At last, 2D plot of the data can be viewed with Visualize tab.

WEKA employs a special file format that is called Arff (Attribute-Relation File Format). Arff comprises 3 parts which are relation, attribute, and data. Relation part contains a word or a string used to assign a name to the dataset. Attribute part, on the other hand, consists of attributes each of which has a name, a data type (which must either enumerated, real or integer data type) and a value range (enumerations for nominal data, intervals for numeric data). The examples of this relation are given in the comma-separated form to make easier interaction with spreadsheets and databases. Figure 3.5 demonstrates an example of an arff file.

```
@RELATION Haber

@ATTRIBUTE köy   REAL
@ATTRIBUTE küçücük   REAL
@ATTRIBUTE fıçıcık   REAL
@ATTRIBUTE işi   REAL
@ATTRIBUTE dolu      REAL
@ATTRIBUTE CLASS {ekonomi,spor}

@DATA
2,3,1,1,1,ekonomi
0,0,0,0,0,ekonomi
1,1,1,1,1,ekonomi
0,0,0,0,0,spor
0,0,0,0,0,spor
0,0,0,0,1,spor
```

Figure 3.5 A view from the Arff file

3.2. Methods

3.2.1. Stemming Algorithms

In text mining and information retrieval, the process of finding root of a word is named as “stemming”. There are lots of stemming algorithms each of which is designed for different languages. Stemming algorithm is language

dependent because it must use appropriate language rules to find roots of terms. In this study we use Zemberek, Affix Striping, and Fixed Prefix algorithms that are developed for Turkish languages.

3.2.1.1. Zemberek

The best part of NLP solutions in the IT world is built on Indo-European languages. The necessity for an extendable NLP library for general purposes arose due to the challenges that inherently exist in the agglutinative languages and the absence of solutions for Turkic languages. Having started with Turkish, Zemberek project now focuses on eliminating this gap and providing a flexible, open source platform-independent NLP framework that will be used not only for Turkish but also for other Turkic languages (Akın and Akın, 2007).

Zemberek is an open source library which has been developed for especially Turkish and other Turkic languages. It has been used orthography control program at Open Office Turkish version and Pardus. There was no any other open source library until Zemberek was generated.

We use Zemberek in this study to find root forms of words. Zemberek makes a morphological breakdown. In order to find the right roots, Zemberek must first determine the roots appropriate to that word. The root selector is used for this purpose. The root selector takes the word like a tree (e.g., a trie), gathers the letters in the word, moves towards the word, and creates the roots that may be appropriate for the word. It picks candidate roots from the roots it collects; and starts the analysis process. Selected roots as candidates are tried one by one, and the words start from the last root to control because the last one is mostly rooted to the correct result. The Zemberek uses a dictionary to find the roots. For each candidate root, the suffixes are added to those roots until they match the first word. For example, for the word “kurul”; the trie structure used by the Zemberek is shown in Figure 3.6.

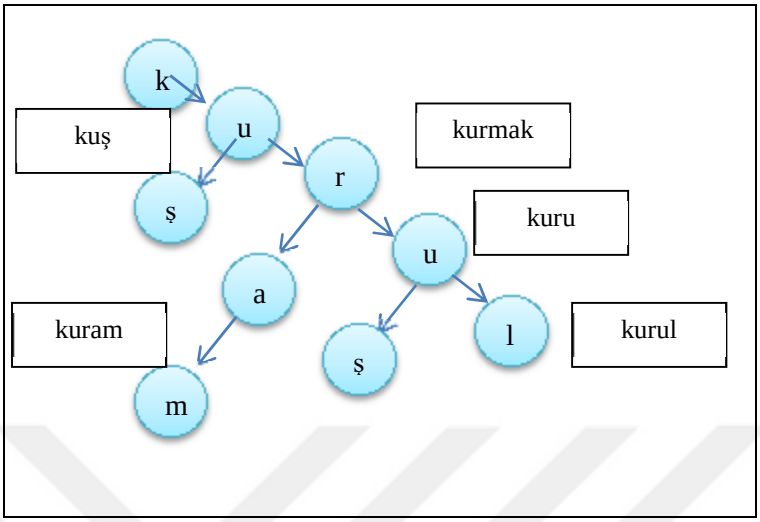

Figure 3.6. An example trie for Zemberek

Beginning with the letter “k” of the word, it collects letters downward. At the same time, it looks at the possible root forms. “k” takes the letter “u” next to it and looks at possible roots, starting with “ku” like “kuş”, then it gets the third letter next to “kur”, it is root of the word “kurmak”. At the same time, it can be obtained from words such as “kura” and “kuram”. When it takes the 4th letter, the word becomes “kuru” and this becomes a meaningful word while the word “kuruş” can also be obtained from this word. Finally, the word “kurul” is formed and compared with possible word roots to the first state of the word. If there is a match with the first word, the right root is reached. It keeps them in a text-based file when looking at word roots and checking the root of words within this file.

3.2.1.2. Affix Stripping

Affix stripping is a different morphological analyzer applied to Turkish. Affix stripping is new methodology proposed for doing the analysis of Turkish words without using any lexicon (Eryiğit and Adalı, 2004). Finite State Machine (FSM) is used in the separation of the roots. In this study, when the words are

separated into their roots, the suffixes are removed one by one from right to left or last to first. For example, when we check the word “kitapçılar” it is analyzed as shown below:

kitapçılar ⟶ kitapçı ⟶ kitap

This process is done by FSM. The FSM can pass finite number of entries from one state to another state and it can produce an output in these states.

Firstly, Turkish suffixes have been classified and then suffixes (i.e., nominal verb suffixes, derivational suffixes, noun suffixes, tense&person verb suffixes, verb suffixes) and their classes have been stored in a database with different tables. In order to produce the results quickly, the structures those the suffixes can take according to the harmonic fit are added to the database by using abbreviations.

When a root is to be found, firstly the FSM of this word is created. The suffixes which can be in the word and the rules which can be applied to these suffixes are collected in an FSM. After the FSM is created, the suffixes of the words are numbered, tables are created with numbered suffixes, then suffix tables are created and these tables are included in to the database. Suffixes are called with these numbers at the created system. For analyzing the words in FSM, the words created at first stage are reversed. When the FSM is reversed, FSM takes an uncertain form. At the next stage if there are any empty states, these states are eliminated, and FSM takes a certain form. At the end, last form of FSM is obtained from right to left with created new states. Reached word with correct transitions is accepted as the root.

3.2.1.3. Fixed Prefix Stemming

The fixed prefix approach is a pseudo stemming technique. In this method, we simply truncate the words and use the first *n* characters of each word as its

stem; words with less than *n* characters are used with no truncation (Can et al., 2008). In this study we use 3, 5, 7 characters as the lengths of the word stems. Let's assume that we want to generate 3, 5, 7 characters long word stems. If our word is "transferlerini", 3, 5, 7 characters long word stems are listed below;

Fixed Prefix 3: "tra"
Fixed Prefix 5: "trans"
Fixed Prefix 7: "transfe"

3.2.2. Stopwords Removal

Stopwords are the words which are filtered out before or after processing of natural language data (text). These words are used frequently (e.g., for Turkish "bir", "şu", "ve", "acaba", "bazen") in languages. It is considered that they will not have any positive effect on the classification because these words occur frequently in documents so they are negligible. There is no any universal stopword list for Natural Language Processing tasks. The stopword list used in this study is obtained from Kaya and Özel (2014) and is given in Appendix 2.

3.2.3. *n*-Gram

n-gram model is generally used in natural language processing. The items in *n*-gram model can be phonemes, syllables, letters, words, or base pairs according to the application. A word *n*-gram is a sequence of *n* words: For a sentence "Please turn your homework" a 2-gram (or bigram) is a two-word sequence of words like "please turn", "turn your", or "your homework", and a 3-gram (or trigram) is a three-words sequence of words like "please turn your", or "turn your homework" (Jurafsky and Martin, 2014). In this study, word *n*-grams that are word 1-2 grams and word 1-2-3 grams are used as features. Word 1-2 grams are the union of word 1-grams and 2-grams, similarly word 1-2-3 grams are the union of word 1-grams, 2-grams, and 3-grams. How to use 1-2 grams and 1-2-3 grams is described below;

Let's assume that we want to generate 1-2 and 1-2-3 grams of the following words sequence: "maçı kurtaran adam". Below 1-2 grams and 1-2-3 grams are listed, and space characters are shown with "_" character.

1-2 grams are: "maçı", "kurtaran", "maçı_kurtaran", "adam", "kurtaran_adam"

1-2-3 grams are: "maçı", "kurtaran", "maçı_kurtaran", "adam", "kurtaran_adam", "maçı_kurtaran_adam".

3.2.4. Term Weighting Methods

Term weighting is a procedure that takes place during the text classification process in order to assess the value of each term in the document. In text classification, each document is expressed with a vector by using Vector Space Model; and each vector element corresponds to the weight of a term in the document. In this way we can express documents as a numerical data and we can use these vectors for classification. In this study *tf*, normalized *tf* (*normtf*), *logtf*, *tp* and *tf*idf* term weighting methods are used.

Term frequency (*tf*) is the observed frequency of terms in the document. It is calculated separately for each term in the document as explained in equation 1.

$$tf_{ij} = frequency\ of\ term\ j\ in\ document\ i \tag{3.1}$$

There are different types of term frequency (*tf*). For example, normalized term frequency (*normtf*) of term j for document i is obtained with the term frequency divided by the document length as in equation 2.

$$normtf_{ij} = \frac{tf_{ij}}{\sum_i tf_{ij}} \tag{3.2}$$

Logtf is obtained by taking the logarithm of the term frequency in base 10 as in equation 3.

$$\log tf_{ij} = log_{10}(tf_{ij}) \tag{3.3}$$

Tp is the term presence in which if the term frequency for term *j* is greater than zero for document *i*, it is equal to 1, otherwise it is equal to 0 (i.e., equation 4).

$$tp_{ij} = \begin{cases} 0, & if\ tf_{ij} = 0 \\ 1, & if\ tf_{ij} > 0 \end{cases} \tag{3.4}$$

The importance of a term for a document collection is calculated by inverse document frequency (*idf*) value. The more documents include a word, the lower the *idf* value of that term. The words such as “and”, “or” etc. are often used frequently in the documents and they are not considered as important to determine class label of the document, so their *idf* values are low. *idf* of a term *j* for the document collection is calculated as follows:

$$idf_j = log \frac{n}{n_j} \tag{3.5}$$

where, n_j is equal to document frequency (*df*) which is the number of documents in the document collection that contain the term *j; n* is the total number of documents in the dataset.

*tf*idf* is one of the most popular weighting method and it is calculated by multiplying the term frequency of term *j* in document *i* with the inverse document frequency of the term *j*. The aim is to give more importance to the words that occur frequently in the document but appear in a small number of documents in the whole collection. This value is computed as shown in equation 6.

$$tf * idf_{ij} = tf_{ij} \times idf_{ij} \quad (3.6)$$

3.2.5. Feature Selection Methods

There are various algorithms to do feature selection on text mining. Purpose of feature selection on text mining is to remove useless terms for classification, therefore to increase accuracy. On the other hands, by removing irrelevant and useless data we decrease the size of datasets and also improve runtime performance of classifiers. In this study, we investigate the effects of information gain and chi square feature selection methods using WEKA tool. Also, we propose a simple feature selection method by computing standard deviations of feature occurrences and compare its performance with the well-known information gain and chi square methods.

3.2.5.1. Using Standard Deviation as Feature Selector

The fields of mathematics and statistics usually make use of the standard deviation method to calculate how data is allocated in connection with the arithmetic mean. The mean of the allocation is used as a reference point by the standard deviation which calculates variability by taking into account the distance between each score and mean. To put it simply, the standard deviation supplies a scale for the standard or average distance from the mean and explains whether the scores are gathered closely around the mean or they are broadly distributed. Equation 7 demonstrates the standard deviation of a variable *x* of a dataset where we have *n* data instances computed.

$$S = \sqrt{\frac{\sum_i^n (x_i - \bar{x})}{n}} \quad (3.7)$$

where *S* is the standard deviation for the variable *x*, $\bar{x}$ is the mean value of the variable *x*, x_i is the i^{th} value of the variable *x*, and *n* is the number of instances in the dataset.

As a feature selector, we use the standard deviation as follows:

In this study, standard deviations of the terms for each class of each datasets that are 1150 News, 3000 Tweet, Turkish Email, 25 Author datasets are calculated. To compute standard deviation of a term for a given dataset, first of all total frequencies of the term for each class in the dataset are computed. Then the standard deviation of these class-based frequencies is computed by using equation 7 for each term extracted from the dataset. Terms are sorted in descending order with respect to their standard deviations, and terms with standard deviation greater than zero are selected, others are eliminated. The top 0.1, 0.5, 1, 5, 10, 15, 20 percent ranked terms are taken, and classification experiments are repeated by using only the selected terms. After that, a classification is also carried out by taking all the terms whose standard deviations are greater than zero.

3.2.5.2. Information Gain

Supposing that we divide the tuples in D on some attribute A having v distinct values, $\{a_1, a_2, \dots, a_v\}$, as observed from the training data. Supposing that A is discrete-valued, these values correspond directly to the v outcomes of a test on A. Attribute A can be employed to partition D into v partitions or subsets, $\{D_1, D_2, \dots, D_v\}$, where D_j includes those tuples in D that have outcome a_j of A. These partitions would correspond to the branches obtained from node N. By preference, an exact classification of the tuples is desired to be produced through this partitioning. In other words, it is desired that each partition is pure. Nonetheless, the partitions will probably be impure (e.g., it is likely that a partition may include a collection of tuples from different classes rather than from a single class). Equation 9 can be applied to calculate how much more information we will need to ensure an exact classification once the partition has been completed.

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} x\ Info(D_j) \tag{3.8}$$

The term $|D_j|/|D|$ behaves as the weight of the j^{th} partition. $Info_A(D)$is the predicted information needed to classify a tuple from D based on the partitioning by A. $Info(D_j)$ is the entropy of dataset D_j which is the subset of D where attribute A assumes a value that is equal to a_j. The smaller the predicted information (still) needed is, the greater the purity of the partitions will be. The difference between the original information need (i.e., based on just the proportion of classes) and the new need (i.e., obtained after partitioning on A) is defined as information gain. That is,

$$Gain(A) = Entropy(D) - Info_A(D) \tag{3.9}$$

In other words, *Gain*(A) tells us how much would be obtained by partitioning on A. This is the predicted reduction in the information need that is caused as a result of knowing the value of A. The attribute A that has the highest information gain, *Gain*(A), is selected as the splitting attribute at node N. We may also express this as such: we want to branch on the attribute A that would do the "best classification," to ensure that the amount of information still needed to end classifying the tuples is least (i.e., minimum $Info_A(D)$)

3.2.5.3. Chi Square (χ^2)

Chi-Square is employed to explore the difference between the observed and predicted values. *Chi-Square* statics are used to compare results for two (or more) independent groups or to compare the results of categorical responses.

Chi-square (χ^2) operates in the following way (Han et al., 2012):

Let's suppose A has c distinct values, namely a_1, a_2,…, a_c . B has r distinct values, namely b_1,b_2,…,b_r . The data tuples defined by A and B can be indicated as a contingency table, with the c values of A constituting the columns, and the r values of B constituting the rows. Let (A_i,B_j) denote the joint event that attributes A

assumes value a_i and attribute B assumes value b_j , that is, where (A=a_i, B=b_j). Each and every possible (A_i,B_j) joint event has its own cell (or slot) in the table. The χ^2 value (also known as the Pearson 2 statistic) is computed as

$$\sum_{i=1}^{c}\sum_{J=1}^{r} \quad \frac{(o_{ij}-e_{ij})^2}{e_{ij}} \tag{3.10}$$

where o_{ij} is the *observed frequency* (i.e., actual count) of the joint event (A_i,B_j) and e_{ij} is the *expected frequency* of (A_i,B_j), which can be computed as

$$e_{ij} = \frac{(count(A=a_i)\ x\ count(B=b_j))}{n} \tag{3.11}$$

where n is the number of data tuples, $count(A = a_i)$ is the number of tuples having value a_i for A, and $count(B = b_j)$is the number of tuples having value b_j for B. The sum in Eq. (12) is calculated over all of the (r x c) cells. We need to keep in mind that the cells that make the most contribution to the χ^2 value are those for which the actual count is very different from that expected.

The χ^2statistic tests the hypothesis that A and B are *independent*, in other words, there is no correlation between them. The test is based on a significance level, with *(r -1) x (c-1)* degrees of freedom.

3.2.6. Naïve Bayes Multinomial Classifier

The Naïve Bayes classification algorithm, which is used to calculate a set of probabilities by way of counting the frequency and combinations of values in a certain dataset, is a simple probabilistic classifier (Patil and Sherekar, 2013). Thomas Bayes (1702-1761), who proposed the Bayes theorem, developed this classification.

Before performing the classification, Naïve Bayes classifier statistically considers high rate and firstly looks at the realization probability. Multinomial Naïve Bayes classifier is aimed at exploring the proportional relationship between data.

The Naïve Bayes classifier operates in the following way (Leung, 2007):

1. Let T be a training set of samples, each with their class labels. There are k classes, $C_1, C_2, \ldots, C_k$. Each sample is represented by an n-dimensional vector, $X = \{x_1, x_2, \ldots, x_n\}$, depicting n measured values of the n attributes, $A_1, A_2, \ldots, A_n$, respectively.

2. The classifier, when given a sample X, will predict that X belongs to the class having the highest posteriori probability, conditioned on X. In other words, X is predicted to belong to the class C_i if and only if

$$P(C_i|X) > P(C_j|X) \text{ for } 1 \leq j \leq m, j \neq i.$$

Thus we obtain the class that maximizes $P(C_i|X)$. The class C_i for which $P(C_i|X)$ is maximized is called the maximum posteriori hypothesis. According to Bayes' theorem, $P(C_i|X)$ is calculated as follows:

$$P(C_i|X) = \frac{P(X|C_i)\,P(C_i)}{P(X)} \tag{3.12}$$

3. Since $P(X)$ is the same for all classes, only $P(X|C_i)P(C_i)$is maximized. If the class probabilities, $P(C_i)$, are not known, then it is generally thought that the classes are equally probable, i.e., $P(C_1) = P(C_2) = \ldots = P(C_k)$, and therefore, we would maximize $P(X|C_i)$. Otherwise, we maximize $P(X|C_i)P(C_i)$. We need to keep in mind that the class probabilities may be predicted through $P(C_i)$ = frequency$(C_i,T)/|T|$.

4. In the case of datasets that possess a large number of attributes, computing $P(X|C_i)$ would be computationally expensive. The Naïve assumption of class conditional independence is made to reduce computation in evaluating $P(X|C_i)$. This makes the assumption that the values of the attributes are conditionally independent of one another in the case of the class label of the sample. To put it in mathematical terms,

$$P(X|C_i) \approx \prod_{k=1}^{n} P(x_k|C_i) \tag{3.13}$$

The training set can be employed to easily predict the probabilities $P(x_1|C_i)$, $P(x_2|C_i)$, . . . , $P(x_n|C_i)$. Note that x_k here refers to the value of attribute A_k for sample X.

(a) If A_k is categorical, then $P(x_k|C_i)$ is the number of samples of class C_i in T having the value x_k for attribute A_k, divided by frequency (C_i, T), the number of samples of class C_i in T.

(b) If A_k is continuous-valued, then we normally presume that the values show a Gaussian distribution with a mean μ and standard deviation σ, as defined by equation 14.

$$g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{(x-\mu)^2}{2\sigma^2} \tag{3.14}$$

so that probability is computed as in equation 15.

$$p(x_k|C_i) = g(x_k, \mu C_i, \sigma Ci) \tag{3.15}$$

μC_i and σC_i, which are the mean and standard deviation of values of attribute A_k , need to be calculated for training samples of class C_i.

$P(X|C_i)P(C_i)$ is evaluated for each class C_i to estimate the class label of X. The classifier predicts that the class label of X is C_i only on condition that it is the class that maximizes $P(X|C_i)P(C_i)$. (Güran et. al, 2009)

The words in a document are considered to be drawn from an underlying multinomial distribution independently of each other in Naïve Bayes Multinomial (NBM) classifier. The number of manifestations (or some weight) of words in the document represents the document. In NBM, the most probable class C_{NBM} for document $d_q = \{w_1, w_2, \dots, w_{|v|}\}$ is calculated through equation 16.

$$C_{NBM} = \underset{C_j \in C}{argmax} \log P(C_j) + \sum_{i=1}^{|V|} f_i \log P(w_i | C_j) \quad (3.16)$$

where the conditional probability $P(w_i|C_j)$ is the relative frequency of the term w_i in documents belonging to the class C_j. McCallum and Nigam (1998) revealed that the Naïve Bayes Multinomial classifier based on the multinomial distribution has a better performance than the multi-variate Bernoulli classifier in terms of text classification applications. The reason why we use Multinomial Naïve Bayes model in this thesis is that it is especially applicable to text classification (Güran et. al, 2009).

In this study, classification success is computed using F-Measure. F-Measure is a scoring, verification concept in statistics science. F-Measure is a measure of the accuracy of the data being tested. In the formulation, it is expressed by precision with p and by recall with r, where p indicates how successful the positive states are predicted, and r is a situation that shows success in a positively predicted situation. Then, F-Measure is the harmonic mean of recall and precision as computed in equation 17.

$$F - Measure = 2\,x\,\frac{precision\ x\ recall}{precision + recall} \quad (3.17)$$

4. RESULTS AND DISCUSSIONS

In this section experiments performed and their results are presented. Perl programming language (http://www.perl.org/) was used for implementing the weighting and the proposed feature selection methods. Preto tool was employed to apply preprocessing methods and compute document vectors. WEKA Data Mining Tool was applied for the classification, and information gain and chi square feature selection methods. The experiments were performed on a computer having Windows 10 operating system, 4 GB of RAM, and Intel Core i5 2.27 GHZ processor.

The experiments consist of seven main parts. The first part covers determination of best term weighting method; the second part includes the effects of stopword usage; the third part investigates which stemming algorithm to be applied; the fourth part tests whether the n-grams should be used or not; the fifth, sixth, and the seventh parts present the effects of feature selection by using standard deviation, information gain, and chi square methods, respectively. Before starting the experiments, number of classes, number of instances, total number of words, number of unique words, number of characters, average characters per word, and average words per document were analyzed for each dataset and these values are shown in Table 4.1.

Table 4.1. Number of classes, number of instances, total number of words, number of unique words and number of characters for each dataset.

Dataset	# of classes	# of instances	Total # of words	# of unique words	# of characters
1150 News	5	1,150	187,638	44,983	1,760,823
3000 Tweet	3	3,000	33,624	10,780	215,123
Turkish Email	2	800	180,793	46,279	1,879,238
25 Author	25	1,250	486,360	94,370	4,434,664

As shown in Table 4.1, there are 5 classes, 1150 instances, 187,638 words, 44,983 unique words and 1,760,823 characters in the 1150 News dataset; there are 3 classes, 3000 instances, 33,624 words, 10,780 unique words and 215,123 characters in 3000 Tweet dataset. Turkish Email dataset has 2 classes, 800 instances, 180,793 words, 46,279 unique words and 1,879,238 characters; and finally, there exists 25 classes, 1250 instances, 486,360 words, 94,370 unique words and 4,434,664 characters in the 25 Author dataset. Average number of characters per word (i.e., average word length), and average number of characters per document are shown in Table 4.2.

Table 4.2. Average number of characters per word (i.e., average word length), and average number of characters per document for each dataset.

Dataset	**Average word length**	**Average document length in terms of characters**
1150 News	9.384	1,531
3000 Tweet	6.398	72
Turkish Email	10.394	2,350
25 Author	9.118	3,548

The average word length for the 1150 news dataset is 9.384 characters, the average document length is 1531 characters; the average word length for the 3000 Tweet dataset is 6.398 characters, the average document length is 72 characters; the average word length for Turkish Email is 10.394 characters, the average document length is 2350 characters; the average word length for the 25 author dataset is 9.118 characters, and the average document length is 3548 characters. As it can be seen from Table 4.1 and 4.2, each dataset has different text characteristics from each other and belongs to different text classification problem domain.

4.1. Determination of the Best Term Weighting Method

In this study firstly, terms are extracted from each dataset separately by using PRETO. Then, their weights are computed by our developed Perl program. Any preprocessing methods like stopword removal or stemming are not applied to any datasets in this first experiment.

5 different term weighting methods that are *tf, logtf, tp, normtf* and *tf*idf* are used to form document vectors and arff files are created for each dataset. Arrf file format is a special format which is used in WEKA. From WEKA software Naïve Bayes Multinomial (NBM) classification method is selected for classifications. NBM is preferred as it is faster than other classification methods and it is used for text classification. For test options, 10 folds cross-validation is applied. Classification results in terms of F-measure values of the experiment is shown in Table 4.3.

Table 4.3. F-Measure values of the NBM classifier for different term weighting methods

	Term Weighting Methods				
Dataset	***tf***	***logtf***	***tp***	***normtf***	***tf*idf***
1150 News	93.2	64.3	92.1	92.9	92.9
3000 Tweet	52.4	27.2	52.2	51.3	48.7
Turkish Email	97.9	94.6	98.1	98.1	97.7
25 Author	51.5	21.7	57.1	46.7	77.1

The best F-Measure values for each dataset are underlined. As seen in Table 4.3, for 1150 News dataset, the best F-Measure value obtained is 93.2% and this value is achieved by using *tf* weighting method. 52.4% is the best F-Measure value computed by using *tf* for 3000 tweet, for Turkish e-mail dataset, 98.1% is the best F-Measure value obtained by using *tp* and *norm tf* and lastly, for 25 Author dataset the best F-Measure value obtained is 77.1% and this value is found by using *tf*idf*. For 1150 News and 3000 Tweet datasets, *tf* method leads to the best F-

Measure value. According to the Table 4.3 generally, *tf* and *tf*idf* term weighting methods are successful for all datasets, and *logtf* has the worst performance. To have the best F-Measure values, in the subsequent experiments both *tf* and *tf*idf* term weighting methods are used.

4.2. Effects of Stopwords Removal

In this experiment, we compare the performance of eliminating stopwords or not from the feature sets for classification accuracy. By using PRETO, stopwords are removed from all documents in the datasets, and the remaining terms are used as features. When the stopwords are removed from the datasets the number of terms in 1150 News decreases by 179 terms, in 3000 Tweet dataset number of terms reduces by 116 terms, in Turkish Email dataset 120 terms are removed, and in 25 Author dataset 171 terms are eliminated. After removing of stopwords, numbers of terms extracted are presented in Table 4.4 where NSNS means No-Stemming No-Stopword Removal, NSWS means No-Stemming With-Stopword Removal.

Table 4.4 Number of terms for each dataset

Dataset	**# of term NSNS**	**# of term NSWS**
1150 News	44,983	44,804
3000 Tweet	10,780	10,664
Turkish Email	46,279	46,159
25 Author	94,370	94,199

After extracting terms with and without stopwords, document vectors are computed by using *tf* and *tf*idf* term weighting methods and arff files are created for each dataset. NBM method is used to classify the datasets. F-measure values of classification with or without stopword removal for all datasets are shown in Table

4.5. The best F-measure values are shown in bold face and underlined, and the second best values are only underlined.

Table 4.5. F-measure values for the datasets with NSNS and NSWS preprocessing

Dataset	NSNS tf	NSNS tf*idf	NSWS tf	NSWS tf*idf
1150 News	93.2	92.9	**93.4**	93.2
3000 Tweet	**52.4**	48.7	52	47.7
Turkish Email	97.5	97.7	**98.6**	97.6
25 Author	51.5	77.1	57.3	**80.5**

As shown in Table 4.5, for 1150 News dataset when the stopwords are removed for *tf* method, F-Measure value increases from 93.2% to 93.4%, for *tf*idf* F-Measure value increases from 92.9% to 93.2% as well. For 3000 Tweet dataset this time F-Measure value decreases for *tf* method from 52.4% to 52% and for *tf*idf* it decreases from 48.7% to 47.7%. In Turkish Email dataset when stopwords are removed, F-Measure increases from 97.5% to 98.6% for *tf* and it decreases from 97.7% to 97.6 % for *tf*idf*. For 25 Author dataset if *tf* weighting method is used, F-measure value increases from 51.5% to 57.3%; for the *tf*idf* weighting method F-Measure value increases from 77.1% to 80.5% when stopwords are removed. 25 Author dataset has the highest increase with respect to other datasets. Only for 3000 tweet *tf*, *tf*idf* and Turkish Email *tf*idf*, F-Measure values decrease when stopwords are removed from the datasets and there has been a slight increase in other F-Measure values. For the subsequent experiments stopwords are removed from the datasets because stopword removal increases F-Measure values in majority of the cases while reducing the feature space.

4.3. Determination of the Best Stemming Algorithm

In this study, 3 different stemming algorithms (i.e. Zemberek, Affix Stripping, Fixed Prefix) are used. Stemming algorithms were discussed in the previous section. Fixed Prefix with 3, 5, 7 characters term lengths are used in this study. By using PRETO, stemming algorithms are applied and also stopwords are removed from the documents. First, Zemberek stemming algorithm is applied to the words in all documents of datasets and root-finding process is carried out. Then, Affix Stripping algorithm and Fixed Prefix 3, 5, 7 algorithms are applied to the datasets. Number of features obtained after applying the stemming algorithms are shown in Table 4.6.

The abbreviations used in Table 4.6 are as follows: The first 2 letters in the first column show the case applied for stemming such that “NS” means “No Stemming”, whereas the next two letters in the first column and first two letters in the other columns “WS” stands for with stopword removal. The last 2 characters show the stemming algorithm used, so that “WZ”, “WA”, “WF” stand for stemming applied with Zemberek, Affix Stripping, and Fixed Prefix Stripping. The first column in Table 4.6 gives the number of tokens extracted when stemming is not applied and stopwords are eliminated.

Table 4.6. Number of unique tokens after applying stemming algorithms

	NSWS	WSWZ	WSWA	WSWF3	WSWF5	WSWF7
1150 News	44,804	11,625	22,588	3,286	13,578	25,302
3000 Tweet	10,664	4,923	6,986	2,030	5,654	8,632
Turkish Email	46,159	11,750	22,848	3,335	13,751	26,463
25 Author	94,199	16,526	40,075	3,655	19,305	43,489

When stemming is applied to the datasets, the number of tokens obtained vary depending on the algorithm used. In Table 4.6, the highest number of tokens

in all datasets are obtained for NSWS. Because no stemming process is applied. When the Fixed Prefix 3 algorithm is applied, the number of unique tokens decreases most. For 1150 News dataset, it decreases from 44,804 to 3,286; for 3000 Tweet dataset it decreases from 10,664 to 2,030; for the Turkish E-mail, it reduces from 46,159 to 13,751; for 25 Author dataset it changes from 94,199 to 3,655 unique tokens.

After the stemming algorithms are applied to the datasets, for each stemming algorithms, *tf* and *tf*idf* term weighting methods are used and arff files are created. A total of 10 arff files for each dataset are created and they are classified by using the NBM classification method. The classification results in terms of F-Measure values for Zemberek, Affix Stripping and Fixed Prefix 3, 5, 7 stemming algorithms are shown in Table 4.7. The best F-measure values for each dataset is underlined in the table below.

Table 4.7. The classification results in terms of F-Measure values for Zemberek, Affix Stripping, Fixed Prefix 3,5,7 stemming algorithms

	1150 News ***tf***	**1150 News** ***tf*idf***	**3000 Tweet** ***tf***	**3000 Tweet** ***tf*idf***	**Turkish Email** ***tf***	**Turkish Email** ***tf*idf***	**25 Author** ***tf***	**25 Author** ***tf*idf***
NSWS	93.4	93.2	52	47.7	98.6	97.6	57.3	80.5
WSWZ	91.4	90.9	49.8	48.7	97	97.1	52.2	68.3
WSWA	90.7	91.6	48.2	48.7	97.1	97.2	41.2	62.4
WSWF3	64.8	68.3	34.8	37	93.5	94.5	28.2	25.2
WSWF5	88.6	88.5	46.5	46.7	95.6	96.1	42.4	56.5
WSWF7	91.8	91.9	50.1	47.6	97.4	97.9	52.2	68.8

In Table 4.7, F-Measure values obtained before stemming and after applying the three stemming algorithms are compared. According to Table 4.7 we observe that the most successful stemmer for all datasets is Fixed Prefix7, on the other hand the worst results for all datasets are obtained with Fixed Prefix3. Fixed

Prefix7, which is a stemming algorithm that takes only the first 7 letters of words, gives the most successful result among the stemming algorithms applied due to the fact that it generates longer word stems, and higher number of stems are obtained. The Fixed Prefix3 algorithm gives the worst results among the stemming algorithms due to the number of the stems obtained are low, and stems are short. Besides, the second best stemming algorithm is Zemberek. Classification with raw forms of the terms is more successful than the stemmed terms for Turkish. This result shows that as Turkish is an agglutinative language, stemming changes word meanings and reduces classification accuracy. Therefore, better stemmers are needed for Turkish. So, in the subsequent experiments stopwords are removed and raw forms of the terms are used.

4.4. Determination of Using N-Grams

In the previous experiments stopword removal process has given successful results so for these experiments stopwords are removed, too. First stopwords are removed from the datasets, then term 1-2 grams and 1-2-3 grams are created. The numbers of term 1-2 grams, and term 1-2-3 grams for each dataset are shown in Table 4.8.

Table 4.8. The number of tokens generated after applying term 1-2grams, and 1-2-3grams feature extraction methods

Dataset	# of terms NSWS	# of terms 1-2grams	# of terms 1-2-3grams
1150 News	44,804	293,846	589,890
3000 Tweet	10,664	36,993	64,451
Turkish Email	46,159	210,913	408,941
25 Author	94,199	594,175	1,237,533

Arff file sizes increase because of large number of features extracted when term 1-2 grams and 1-2-3 grams are used as features. As WEKA does not operate with big arff files, file sizes have been decreased by filtering terms whose lengths are less than or equal to 3 characters and having document frequency less than 3. After filtering the terms, the numbers of tokens remained are shown in Table 4.9.

Table 4.9. The number of tokens left after filtering is applied to 1-2grams and 1-2-3grams forms

Dataset	**# of terms NSWS**	**# of terms 1-2grams**	**# of terms 1-2-3grams**
1150 News	44,804	15,509	16,384
3000 Tweet	10,664	2,037	2,170
Turkish Email	46,159	19,667	29,387
25 Author	94,199	34,403	36,154

Then by applying *tf* and *tf*idf* term weighting methods and by using filtered term 1-2 grams and 1-2-3 grams features, arff files are created. Totally 4 arff files are created for each dataset. Datasets are classified by using NBM in WEKA. After the classification, F-Measure values are shown in Table 4.10.

Table 4.10. F-Measure values for term n-grams

Dataset	**NSWS tf**	**NSWS tf*idf**	**1-2 grams tf**	**1-2-3 grams tf**	**1-2 grams tf*idf**	**1-2-3 grams tf*idf**
1150 News	93.4	93.2	93.7	93.7	93.5	93.5
3000 Tweet	52	47.7	52	52	50.7	50.8
Turkish Email	98.6	97.6	97.4	97.4	97.5	97.4
25 Author	57.3	80.5	71.5	71.3	86.7	86.2

The best F-measure values are underlined in Table 4.10. According to the table, with 93.7% F-Measure value, 1-2 grams *tf* and 1-2-3 grams *tf* methods are more successful than other methods for 1150 News dataset; with 52% F-Measure value NSWS *tf*, 1-2 grams *tf* and 1-2-3 grams *tf* methods are more successful for 3000 Tweet dataset; for Turkish Email dataset NSWS *tf* method is more successful with 98.6% F-Measure value than other 5 methods; for 25 Author dataset with 86.7% F-Measure value 1-2 grams *tf*idf* is more successful than other methods. The largest classification accuracy increase is observed for 25 Author dataset in Table 4.10. When the NSWS *tf* method is used for this dataset, F-Measure value is 57.3%, while 1-2 grams *tf*idf* method is applied, F-Measure value rises to 86.7%. For the other datasets, using n-grams does not lead to significant classification accuracy improvement.

4.5. Effects of Using Standard Deviation Based Feature Selection

The experiments which are made with term 1-2 grams and 1-2-3 grams give more successful results for 25 Author dataset, only a slight improvement for 1150 News dataset, but classification accuracy does not improve for 3000 Tweet and Email datasets. The most successful results for all datasets are achieved as a result of experiments with the raw forms of the terms. In the experiments with standard deviation based feature selection, the terms are used in their raw forms. Stopwords have been removed from the datasets because they give better results in experiments with the raw states and n-grams are not used.

The standard deviation-based feature selection is done by calculating the standard deviation of total frequency of each term for each class in the datasets. After calculating the standard deviation of terms, the terms with zero standard deviation are eliminated, and the standard deviations of the remaining terms are sorted in descending order. Classification experiments are done primarily by selecting the top ranked *n* percent features from the sorted features list such that *n* takes values as 0.1, 0.5, 1, 5, 10, 15, and 20, then the classification is also done by

taking all the terms with a standard deviation greater than zero. Document vectors are created by calculating *tf* and *tf*idf* values of each selected term in the documents, and then they are classified by using NBM in WEKA. The numbers of features obtained are given in Table 4.11 after selecting features having standard deviation that is greater than zero, where SD means Standard Deviation.

Table 4.11. The number of terms whose standard deviation is greater than zero

Dataset	**# of terms SD>0**
1150 News	32,275
3000 Tweet	7,806
Turkish Email	33,063
25 Author	70,467

After computing standard deviation values, terms having standard deviation greater than zero are sorted with respect to these values in descending order, then top ranked 0.1 percent terms are selected, and the datasets are classified with respect to these terms. Number of terms selected by this method is given in Table 4.12.

Table 4.12. The number of terms selected after taking top ranked 0.1 percent features having standard deviation greater than zero

Dataset	**# of term 0.1% SD**
1150 News	35
3000 Tweet	8
Turkish Email	33
25 Author	70

After selecting the features, arff files are created for *tf* and *tf*idf* values of each of the selected terms in the files. Then NBM classifier is applied and F-Measure values obtained are given Table 4.13. The best F-measure values are underlined in the table.

Table 4.13. F-Measure values for classification with the selected features by using top ranked 0.1 percent features

	0.1% SD tf	**0.1% SD tf*idf**	**NSWS tf**	**NSWS tf*idf**
1150 News	54.8	57.5	93.4	93.2
3000 Tweet	32.2	32.1	52	47.7
Turkish Email	93.6	93.9	98.6	97.6
25 Author	50.1	50.1	57.3	80.5

When evaluating the classification results obtained by using the top ranked 0.1% features with respect to their standard deviation values we observed that, F-Measure values of 1150 News NSWS *tf* decreases from 93.4% to 54.8%, and value of NSWS *tf*idf* decreases from 93.2% to 57.5%. For 3000 Tweet dataset F-Measure, value of NSWS *tf* reduces from 52% to 32.2%, also F-Measure value of NSWS *tf*idf* reduces from 47.7% to 32.1%. Similarly, F-Measure value for NSWS *tf* of Turkish Email dataset decreases from 98.6% to 93.6%, and F-Measure NSWS *tf*idf* decreases from 97.6% to 93.9%. NSWS *tf* value for 25 Author dataset decreases from 57.3% to 50.1% and also results for NSWS *tf*idf* value decreases from 80.5% to 50.1%. As it can be seen from Table 4.13, all F-measure values for classification reduces when feature selection is applied. Therefore, we need to select more features, so we increase the percentage of the selected features to find better classification results.

Then we select top ranked 0.5 percent features having standard deviations that are greater than zero. The numbers of selected features are given in Table 4.14.

Table 4.14. The number of terms selected after taking top ranked 0.5 percent features having standard deviation greater than zero

Dataset	**# of term 0.5% SD**
1150 News	176
3000 Tweet	39
Turkish Email	165
25 Author	352

After selecting the features, arff files are created for *tf* and *tf*idf* weight values of each of the selected term in the files. Then NBM classifier is applied and F-Measure values obtained are given Table 4.15 where the best F-Measure values are underlined.

Table 4.15. F-Measure values for classification with the selected features by using top ranked 0.5 percent features

Dataset	**0.5 % SD tf**	**0.5 % SD tf*idf**	**NSWS tf**	**NSWS tf*idf**
1150 News	80.2	79.8	93.4	93.2
3000 Tweet	43.2	43.4	52	47.7
Turkish Email	94.6	94.6	98.6	97.6
25 Author	76.2	78.3	57.3	80.5

When evaluating the classification results with top ranked 0.5 percent features with respect to their standard deviations, we observed that F-Measure values of 1150 News dataset for NSWS *tf* decreases from 93.4% to 80.2%, and F-Measure value of NSWS *tf*idf* decreases from 93.2% to 79.2%. For 3000 Tweet dataset, F-Measure value of NSWS *tf* reduces from 52% to 43.2%, also F-Measure value of NSWS *tf*idf* reduces from 47.7% to 43.4%. F-Measure value of NSWS *tf* method for Turkish Email dataset reduces from 98.6% to 94.6%, and for NSWS

*tf*idf* F-Measure value decreases from 97.6% to 94.6%. Only the F-Measure value of NSWS *tf* for 25 Author dataset increases from 57.3% to 76.2% when feature selection is applied. There is a slight decrease in F-Measure for 25 Author dataset when feature selection is applied to NSWS with *tf*idf*. As it can be seen from Table 4.15, all F-measure values for classification reduces except for 25 Author dataset when feature selection is applied. Therefore, we need to select more features, and we increase the percentage of the selected features to find better classification results.

Then we select top ranked 1 percent features having standard deviations that are greater than zero. The number of selected features is given in Table 4.16.

Table 4.16. The number of terms selected after taking top ranked 1 percent features having standard deviation greater than zero

Dataset	**# of term 1% SD**
1150 News	353
3000 Tweet	78
Turkish Email	331
25 Author	705

After selecting the features, arff files are created for *tf* and *tf*idf* weighting values of each of the selected terms in the files. Then NBM classifier is applied and F-Measure values obtained are given Table 4.17 where the best F-Measure values are underlined.

Table 4.17. F-Measure values for classification with the selected features by using top ranked 1 percent features

	1% SD tf	1% SD tf*idf	NSWS tf	NSWS tf*idf
1150 News	85.9	87.9	93.4	93.2
3000 Tweet	45.2	45.9	52	47.7
Turkish Email	94.6	94.2	98.6	97.6
25 Author	68.1	90	57.3	80.5

When evaluating the classification results with top ranked 1 percent features with respect to their standard deviations, we observed that F-Measure values of 1150 News NSWS *tf* decreases from 93.4% to 85.9%, and F-Measure value of NSWS *tf*idf* decreases from 93.2% to 87.9%. For 3000 Tweet dataset, F-Measure value of NSWS *tf* reduces from 52% to 45.2%, also F-Measure value of NSWS *tf*idf* reduces from 47.7% to 45.9%. For Turkish Email dataset, F-Measure values reduces from 98.6% to 94.6%, from 97.6% to 94.2% for NSWS *tf* and NSWS *tf*idf*, respectively. Only for 25 Author dataset F-Measure values increase from 57.3% to 68.1% and also from 80.5% to 90%, for both NSWS *tf* and *tf*idf*, respectively. As it can be seen from Table 4.17, F-measure values for 1150 News, 3000 Tweet, Turkish Email datasets reduce, while F-measure value for 25 Author increases when feature selection is applied. Therefore, we need to select more features, and we increase the percentage of the selected features to find better classification results.

Then we select top ranked 5 percent features having standard deviations that are greater than zero. The numbers of the selected features are given in Table 4.18.

Table 4.18. The number of terms selected after taking top ranked 5 percent features having standard deviation greater than zero

Dataset	# of term 5% SD
1150 News	1,764
3000 Tweet	390
Turkish Email	1,653
25 Author	3,523

After selecting the features, arff files are created for *tf* and *tf*idf* weight values of each selected terms in the files. Then NBM classifier is applied and F-Measure values obtained are given Table 4.19. The best results are underlined in the below table.

Table 4.19. F-Measure values for classification with the selected features by using top ranked 5 percent features

Dataset	5% SD tf	5% SD tf*idf	NSWS tf	NSWS tf*idf
1150 News	91	92	93.4	93.2
3000 Tweet	53.6	54.1	52	47.7
Turkish Email	97	97	98.6	97.6
25 Author	87.8	90	57.3	80.5

As it can be seen from Table 4.19, F-measure values for 1150 News and Turkish Email datasets reduces, F-measure values for 3000 Tweet and 25 Author datasets increase when feature selection is applied. Also, reduction in F-Measure values for 1150 News and Turkish Email dataset are not high as it is observed from Table 4.19.

Then we increase the number of features selected, and we take the top ranked 10 percent features having standard deviations that are greater than zero. The numbers of selected features are given in Table 4.20.

Table 4.20. The number of terms selected after taking top ranked 10 percent features having standard deviation greater than zero

Dataset	**# of term 10% SD**
1150 News	3,528
3000 Tweet	781
Turkish Email	3,306
25 Author	7,047

After selecting the features, arff files are created for *tf* and *tf*idf* weight values of each selected terms in the files. Then NBM classifier is applied and F-Measure values obtained are given Table 4.21 where the best F-Measure values are underlined.

Table 4.21. F-Measure values for classification with the selected features by using top ranked 10 percent features

	10% SD tf	**10% SD tf*idf**	**NSWS tf**	**NSWS tf*idf**
1150 News	91.7	92.3	93.4	93.2
3000 Tweet	56.1	57	52	47.7
Turkish Email	97.6	97.9	98.6	97.6
25 Author	86.6	91	57.3	80.5

As it can be seen from Table 4.21, F-measure values for 1150 News and Turkish Email datasets reduces, F-measure values for 3000 Tweet and 25 Author datasets increase when feature selection is applied. To select more features, we

increase the percentage of the selected features to find better classification results for 1150 News and Turkish Email datasets.

Then we select top ranked 15 percent features having standard deviations that are greater than zero. The numbers of selected features are given in Table 4.22.

Table 4.22. The number of terms selected after taking top ranked 15 percent features having standard deviation greater than zero

Dataset	**# of term 15% SD**
1150 News	5,291
3000 Tweet	1,171
Turkish Email	4,959
25 Author	10,570

After selecting the features, F-Measure values obtained for classification are given in Table 4.23 where the best results are underlined.

Table 4.23. F-Measure values for classification with the selected features by using top ranked 15 percent features

Dataset	**15% SD tf**	**15% SD tf*idf**	**NSWS tf**	**NSWS tf*idf**
1150 News	91.4	91.6	<u>93.4</u>	<u>93.2</u>
3000 Tweet	<u>57.4</u>	<u>59.9</u>	52	47.7
Turkish Email	97.6	<u>98.4</u>	<u>98.6</u>	97.6
25 Author	<u>84.8</u>	<u>91.1</u>	57.3	80.5

When evaluating the classification results with top ranked 15 percent features with respect to their standard deviations, we observed that F-Measure values of 1150 News dataset decrease from 93.4% to 91.4%, and from 93.2% to 91.6% for NSWS *tf* and *tf*idf*, respectively. However classification F-Measure

values increase significantly for all other datasets except for Turkish Email dataset with *tf* weighting.

Therefore, to investigate whether classification accuracy can be increased or not for 1150 News dataset, we continue our experiments by selecting more features. We select top ranked 20 percent features having standard deviations that are greater than zero. The numbers of selected features are given in Table 4.24.

Table 4.24. The number of terms selected after taking top ranked 20 percent features having standard deviation greater than zero

Dataset	**# of term 20% SD**
1150 News	7,055
3000 Tweet	1,561
Turkish Email	6,613
25 Author	14,093

After selecting the features, the classification F-Measure values are presented in Table 4.25, where the best values are underlined.

Table 4.25. F-Measure values for classification with the selected features by using top ranked 20 percent features

	20% SD tf	**20% SD tf*idf**	**NSWS tf**	**NSWS tf*idf**
1150 News	91.4	91.8	<u>93.4</u>	93.2
3000 Tweet	<u>57.6</u>	<u>59.3</u>	52	47.7
Turkish Email	97.7	<u>98.4</u>	<u>98.6</u>	97.6
25 Author	<u>81.3</u>	<u>90.7</u>	57.3	80.5

When evaluating the classification results with top ranked 20 percent features with respect to their standard deviations, we observed that F-Measure

values of 1150 News NSWS *tf* decreases from 93.4% to 91.4%, and value for NSWS *tf*idf* decreases from 93.2% to 91.8%. For 3000 Tweet dataset this value for NSWS *tf* increases from 52% to 57.6%, also F-measure value of NSWS *tf*idf* increases from 47.7% to 59.3%. The results for NSWS *tf* of Turkish Email dataset decrease from 98.6% to 97.7%, and for NSWS *tf*idf* this value increases from 97.6% to 98.4%. F-measure values for NSWS *tf* and NSWS *tf*idf* for 25 Author dataset increase from 57.3% to 81.3%, and from 80.5% to 90.7%, respectively. As it can be seen from Table 4.25, F-measure values for 1150 News and Turkish Email datasets reduce, while F-measure values for 3000 Tweet and 25 Author datasets increase when feature selection is applied. To select more features, we increase the percentage of the selected features.

Then, we select all features having standard deviations that are greater than zero. The numbers of selected features are given in Table 4.11. There are 32,275 features for 1150 News, 7,806 features for 3000 Tweet, 33,063 features for Turkish Email, and 70,467 features for 25 Author datasets.

After selecting the features, arff files are created for *tf* and *tf*idf* values of each of the selected terms in the files. Then NBM classifier is applied and F-Measure values obtained are given in Table 4.26 where the best values are underlined.

Table 4.26. F-Measure values for classification when all features having standard deviation greater than zero are used

	SD tf	**SD tf*idf**	**NSWS tf**	**NSWS tf*idf**
1150 News	93	93	93.4	93.2
3000 Tweet	52.9	47.3	52	47.7
Turkish Email	98.5	97.1	98.6	97.6
25 Author	58.8	81.7	57.3	80.5

When evaluating the classification results with all features having standard deviation greater than zero we observed that, F-Measure values of 1150 News

NSWS *tf* decreases from 93.4% to 93%, and this value for NSWS *tf*idf* decreases from 93.2% to 93%. For 3000 Tweet results, the F-Measure value for NSWS *tf* increases from 52% to 52.9%, also value for NSWS *tf*idf* decreases from 47.7% to 47.3%. F-Measure obtained for NSWS *tf* of Turkish Email dataset decreases from 98.6% to 98.5%, and for NSWS *tf*idf* this value decreases from 97.6% to 97.1%. F-Measure values of NSWS *tf* and NSWS *tf*idf* for 25 Author dataset increase from 57.3% to 58.8%, and from 80.5% to 81.7%, respectively. As it can be seen from Table 4.25, F-measure values for 1150 News, 3000 Tweet, and Turkish Email datasets reduce, while F-measure values for 25 Author increase when feature selection is applied by using all features having standard deviation greater than zero.

When we summarize the experimental results obtained from the feature selection using the standard deviation based method, the number of terms selected are listed in Table 4.27.

Table 4.27. Number of terms selected for each dataset when standard deviation based feature selection is applied

Feature Selection Method Applied	**1150 News**	**3000 Tweet**	**Turkish Email**	**25 Author**
No Selection (NSWS)	44,804	10,664	46,159	94,199
Terms with SD>0	32,275	7,806	33,063	70,467
Top 20% SD terms	7,055	1,561	6,613	14,093
Top 15% SD terms	5,291	1,171	4,959	10,570
Top 10% SD terms	3,528	781	3,306	7,047
Top 5% SD terms	1,764	390	1,653	3,523
Top 1% SD terms	353	78	331	705
Top 0.5% SD terms	176	39	165	352
Top 0.1% SD terms	35	8	33	70

As shown in Table 4.27, the number of features selected decreases as we apply standard deviation-based feature selection.

In Table 4.28, we summarize the F-measure values obtained when *tf* weighting is used, and feature selection is applied. The best F-Measure values obtained are written in boldface and underlined in Table 4.28.

Table 4.28. F-Measure values obtained when standard deviation-based feature selection is applied and *tf* weighting is used

Feature Selection Method	**1150 News**	**3000 Tweet**	**Turkish Email**	**25 Author**
NSWS tf	**93.4**	52	**98.6**	57.3
SD>0 tf	93	52.9	98.5	58.8
20% SD tf	91.4	**57.6**	97.7	81.3
15% SD tf	91.4	57.4	97.6	84.8
10% SD tf	91.7	56.1	97.6	86.6
5% SD tf	91	53.6	97	**87.8**
1% SD tf	85.9	45.2	94.6	68.1
0.5% SD tf	80.2	43.2	94.6	76.2
0.1% SD tf	54.8	32.2	93.6	50.1

As it can be seen from Table 4.28, when *tf* weighting method is used, standard deviation based feature selection improves classification accuracy only for 3000 Tweet and 25 Author datasets. The improvement for 3000 Tweet is small (i.e., from 52% to 57.6%), however for 25 Author dataset this improvement is significant (i.e., from 57.3% to 87.8%) with only selecting the 5% of the features. On the other hands, feature selection for 1150 News and Turkish Email datasets does not increase classification accuracy, however when all features having standard deviation greater than zero are used, the classification F-measure values

do not reduce significantly (i.e., only 0.4% and 0.1% reductions for 1150 News and Turkish Email datasets respectively).

When *tf *idf* weighting is used, F-Measure values obtained before and after the standard deviation based feature selection is applied are listed in Table 4.29, where the best values are written in boldface and underlined.

Table 4.29. F-Measure values obtained when standard deviation-based feature selection is applied and *tf *idf* weighting is used

Feature Selection Method	1150 News	3000 Tweet	Turkish Email	25 Author
NSWS tf*idf	**93.2**	47.7	97.6	80.5
SD>0	93	47.3	97.1	81.7
20% SD tf*idf	91.8	59.3	**98.4**	90.7
15% SD tf*idf	91.6	**59.9**	**98.4**	**91.1**
10% SD tf*idf	92.3	57	97.9	91
5% SD tf*idf	92	54.1	97	90
1% SD tf*idf	87.9	45.9	94.2	90
0.5% SD tf*idf	79.8	43.4	94.6	78.3
0.1% SD tf*idf	57.5	32.1	93.9	50.1

According to Table 4.29, classification F-Measure values improves for all datasets except 1150 News when standard deviation based feature selection is applied. The most significant improvements are observed for the 25 Author and 3000 Tweet datasets. For 1150 News dataset, applying standard deviation based feature selection only reduces F-Measure value for 0.2%. As a result of this experiment we can conclude that, using standard deviation based feature selection either improves or does not reduce significantly the classification F-measure value by reducing the feature space more than 50%. Our proposed feature selection has better performance when *tf*idf* weighting is used and applied to datasets having

large number of classes. When number of classes increase computing standard deviation for terms becomes more meaningful to show the differences among distinct classes.

4.6. Effects of Using Information Gain as Feature Selector

After measuring the effects of using standard deviation-based feature selection, we apply Information Gain (IG) as feature selector to compare with our proposed SD based feature selection method. In this study, after calculating the IG values of terms, features are selected by taking terms that have scores greater than zero. The number of terms whose information gain value is greater than zero are given in Table 4.30

Table 4.30. Number of terms selected after applying IG based feature selection

Dataset	# of terms
1150 News	1890
3000 Tweet	144
Turkish Email	4671
25 Author	210

F-Measure values after IG based feature selection applied are compared with the results of the most successful standard deviation based feature selection method and with the no feature selection (NSWS) case. The experimental results are shown in Table 4.31 where the best values are underlined.

Table 4.31. Comparison of SD and IG methods with no feature selection case

Dataset	SD tf	SD tf*idf	NSWS tf	NSWS tf*idf	IG tf	IG tf*idf
1150 News	93	93	93.4	93.2	92.1	92.3
3000 Tweet	57.6	59.9	52	47.7	43.8	44
Turkish Email	98.5	98.4	98.6	97.6	97.4	97.2
25 Author	84.8	91.1	57.3	80.5	67.2	40

When the obtained F-measure values are examined, the feature selection with the Information Gain method has not shown a successful classification performance in any dataset. According to the table, the most successful F-measure value for *tf* weighting for 1150 News dataset decreases from 93.4% to 92.1%, for *tf*idf* weighting F-measure value decreases from 93.2% to 92.3% when IG based feature selection is applied. Similarly, the best F-measure value for *tf* weighting for 3000 Tweet dataset reduces from 57.6% to 43.8%, for *tf*idf* weighting F-measure value reduces from 59.9% to 44% after applying IG feature selection. Similar reduction in F-measure values for other datasets are also observed when IG based feature selection is applied. When the two feature selection methods are compared we can say that standard deviation based method outperforms IG based method for all the datasets especially for the 25 Author dataset where the most successful F-Measure value for *tf* weighting is 84.8% for SD method, it reduces to 67.2% for IG method, for *tf*idf* weighting this value decreases from 91.1% to 40% when IG feature selector is applied. As the number of features having IG score greater than zero is not high, we do not apply any further filtering by selecting top ranked *n* percent of the features. Further experimental analysis may be performed by selecting various numbers of features with respect to their IG scores, or both feature selection methods can be combined with a search algorithm to get optimal number of features.

4.7. Effects of Using *Chi Square* (χ^2) Measure as Feature Selector

After measuring the effects of using standard deviation-based feature selection, we apply *Chi Square* measure as feature selector to compare with standard deviation based feature selection method. In this study, after calculating the chi square values for each term, features are selected by taking terms that have chi square scores greater than zero. The number of terms whose chi square value is greater than zero are presented in Table 4.32.

Table 4.32. Number of terms selected after applying *Chi Square* based feature selection

Dataset	# of terms
1150 News	1889
3000 Tweet	144
Turkish Email	4671
25 Author	208

F-Measure values after *Chi Square* based feature selection is applied are compared with the results of the most successful standard deviation based feature selection method and with the no feature selection case (i.e., NSWS method). The experimental results are shown in Table 4.33 where the best results are underlined.

Table 4.33. Comparison of SD and Chi Square methods with no feature selection case

Dataset	SD tf	SD tf*idf	NSWS tf	NSWS tf*idf	Chi square tf	Chi square tf*idf
1150 News	93	93	93.4	93.2	92.1	92.3
3000 Tweet	57.6	59.9	52	47.7	43.8	44
Turkish Email	98.5	98.4	98.6	97.6	97.4	97.2
25 Author	87.8	91.1	57.3	80.5	67.2	40

When the obtained F-measure values in Table 4.33 are examined, the feature selection with the *Chi Square* method has not shown the most successful classification performance in any dataset and this method has very similar performance with that of IG feature selection. According to the table, the most successful F-measure value for *tf* weighting for 1150 News dataset decreases from 93.4% to 92.1%, for *tf*idf* weighting F-measure value decreases from 93.2% to 92.3% when *Chi Square* based feature selection is applied. Similarly, the best F-measure value for *tf* weighting for 3000 Tweet dataset reduces from 57.6% to 43.8%, for *tf*idf* weighting F-measure value reduces from 59.9% to 44% after applying *Chi Square* feature selection. Similar reduction in F-measure values for

other datasets are also observed when *Chi Square* based feature selection is applied. When the two feature selection methods are compared we can say that standard deviation based method outperforms *Chi Square* based method for all the datasets especially for the 25 Author dataset where the most successful F-Measure value for *tf* weighting is 84.8% for SD method, it reduces to 67.2% for *Chi Square* method, for *tf*idf* weighting this value decreases from 91.1% to 40% when *Chi Square* feature selector is applied. As the number of features having *Chi Square* score greater than zero is not high, we do not apply any further filtering by selecting top ranked *n* percent of the features. Further experimental analysis may be performed by selecting various numbers of features with respect to their *Chi Square* scores, or both feature selection methods can be combined with a search algorithm to get optimal number of features.

4.8. Effects of the Feature Selection to the Classification Time

As a result of the study, the most successful methods are *tf* and *tf*idf* among the term weighting methods, and all experiments are done with these methods. We compare the effect of the feature selection to the time required for classification when *tf* weighting is used, and the results in terms of seconds are given Table 4.34.

Table 4.34. Time Required (in seconds) for Classification when the Best F-Measure Values Obtained

	NSWS Values			**Best Feature Selection Values**		
Dataset	**# of terms**	**Time**	**F-Measure values**	**# of terms**	**Time**	**F-Measure values**
1150 News	44,804	15	93.4	15,509	7	93.7
3000 Tweet	10,664	7	52	1,561	1	57.6
Turkish Email	46,159	8	98.6	33,063	7	98.5
25 Author	94,199	94	57.3	3,523	9	87.8

When the numbers of features are reduced by applying feature selection, time required for classification also reduces in proportional to the number of features reduced as it is expected. F-Measure values also improve when feature selection is applied. Therefore, feature selection allows us to make faster classification without reducing classification accuracy for 1150 News and Turkish Email datasets, and with increasing classification accuracy for 25 Author and 3000 Tweet datasets.

4.9. Comparison with Earlier Studies and Discussions

Tunalı and Bilgin (2012), used NTV and Milliyet newspaper datasets for document clustering. They applied Zemberek, Affix Stripping and Fixed Prefix 3, 5, 7 stemmers, and *tf*idf* term weighting method. Also they removed stopwords from documents. For clustering they used Spherical K-Means algorithm. In our study, we applied 5 different term weighting methods: *tf, tf*idf, normtf, tp, logtf.* We used Naïve Bayes Multinomial (NBM) classifier and we removed stopwords. For stemming the terms, we used Zemberek, Affix Stripping and Fixed Prefix 3, 5, 7, too. In the study conducted by Tunalı and Bilgin (2012) have observed the best results by using Zemberek for the Milliyet dataset. According to our results, Fixed Prefix7 has better performance than Zemberek. This result may occur due to the fact that we applied classification, Tunalı and Bilgin (2012) performed clustering; and also our datasets are different.

Amasyalı and Diri (2006), used Turkish newspaper dataset to make classification with respect to author of the documents, genre of the documents, and gender of the authors. They used character n-gram (bi-gram and trigram) for text categorization. For classification of documents they used Naive Bayes, Support Vector Machine (SVM), C4.5, Relationship-Based Object Selection (CFS), and Random Forests (RF) methods. The best classification method is Naive Bayes for author identification and the best accuracy is 83%, for genre identification SVM is the best classification method and the best accuracy is 93.6%, and gender

identification is done with 96.3% accuracy, and the best classification method is SVM. In our experiments we used 1-2grams and 1-2-3grams of words and Naive Bayes Multinomial (NBM) classifier. We also weighted the documents with *tf* and *tf*idf* term weighting methods. The best classification accuracy is 93.7% in 1150 News dataset and this result is observed with the use of 1-2grams and 1-2-3grams with *tf* method. Our experimental analysis have showed that NBM classification method and words 1-2grams or 1-2-3grams affect the classification accuracy positively.

Yıldız et al. (2007) have made a classification by creating a new feature vector. They used Sabah, Vatan, Hürriyet newspaper datasets. They have used Zemberek for stemming; and Naive Bayes (NB), Support Vector Machine (SVM), C 4.5, K-Nearest Neighborhood (KNN), and Random Forests (RF) for classification. The best accuracy observed was the 96% with Naïve Bayes classifier. In our study, we also applied NBM classifier and compared 5 different term weighting methods. We found *tf* and *tf*idf* weighting have the best performance. After the feature selection was applied, the results improved and the most successful result increased to 98.4%.

Çataltepe et al. (2008) have used 2 datasets as Milliyet and Vikipedi. Each dataset has 5 different categories. Zemberek has been used for stemming of the words. After finding the roots, then they made the choice of these roots with respect to (longest, shortest, first k letters,etc), and not all of the words from each document were used; only the most frequent 20% and 5% terms were taken. Centroid and Support Vector Machine classification algorithms and *tf*idf* term weighting method have been applied in the study. It is observed that the most successful classification accuracy obtained is 74.9% when Zemberek stemmer, and SVM classifier are applied. Also they found that when only 2 or 3 letters long roots are used, the classification performance decreases. In our study, we used Naïve Bayes Multinomial classifier; Zemberek, Affix Stripping and Fixed Prefix 3, 5, 7 to find the word stems. The most successful F-measure value we have achieved when

Zemberek stemmer applied is 91.4% by the *tf* term weighting method. We also showed that short word stems decrease classification performance.

Güran et al. (2009) used *tf*idf* vector space model and PC-KIMMO for obtaining roots on 600 documentary and 6 categories dataset. They also used n-grams (unigram, bigram, trigram) of words. Naive Bayes, Complement Naive Bayes, Multinomial Naive Bayes, C4.5 Decision Tree, K-Nearest Neighbor classification methods are used in their studies. The best classification method is found as Naive Bayes Multinomial with the accuracy of 94.67% on the unigram of words. We used Zemberek, Affix Stripping and Fixed prefix 3,5,7. In our experiment Multinomial Naive Bayes classifier has 93.5% accuracy on the 1-2grams and 1-2-3grams with *tf*idf* weighting. We also found that, term unigram forms with Naive Bayes Multinomial classifier is more successful than terms 1-2grams and 1-2-3grams forms except for the 25 Author dataset.

Torunoğlu et al. (2011) used Hurriyet and Milliyet datasets. They applied binary weighting (*tp*) and term frequency (*tf*) weighting for classification. Naïve Bayes, Naïve Bayes Multinomial, Support Vector Machine, and K-Nearest Neighbor classification methods were used by the researchers. Zemberek and Fixed Prefix 3, 5, 7 stemming algorithms were employed in the experiments. They also used stopword filtering. Fixed Prefix 5 results were better than other stemming algorithms. Best results for classification were observed when Naïve Bayes Multinomial and Support Vector Machine classifiers are used with *tf* weighting scheme. They concluded that stemming and stopword filtering has very little impact. When we compare the results of our experiments with this study, we also found that stemming and stopword filtering have very little impact, and *tf* has better performance than *tp*. Fixed Prefix 7 is the best stemmer; we take first 7 characters of words. Using whole words yields better classification accuracy than using root forms of words.

A comprehensive research about text classification is made by the Amasyalı et al. (2012). Various terms weighting methods and 6 different datasets

are used in this study. 14 different text representation methods were compared and Zemberek library was employed for finding word stems. They have also taken the 2-3grams of letters and 2grams of words. K-Nearest Neighbor, Decision Tree-C4.5, Support Vector Machine, Naïve Bayes and Random Forest classification methods were used. In their study results of n-grams have been more successful than other methods. N-gram weighting with binary, log, and n1 gave better results than the other methods. In our study, the results of 1-2grams and 1-2-3 grams for the 1150 news dataset were more successful than those of other methods.

Uysal and Serkan (2013) used Turkish and English Email datasets, and applied 4 preprocessing methods that are tokenization, stopword removal, lowercase conversion and stemming. They used Zemberek and Fixed Prefix stemming algorithms for Turkish. For classification, Support Vector Machine was utilized and maximum classification accuracy observed was 97.1% with nonstemming and non stopword removal. When they applied stemming, accuracy decreased from 97.1% to 89.1%. In our experiments, when we do not apply stemming and do not remove stopwords classification accuracy is 98.1%, and when we apply stemming and stopword removal the accuracy score decreases from 98.1% to 97.1% for Turkish Email dataset. This result shows that Naïve Bayes Multinomial classification method performs better with stemming and stopword removal than Support Vector Machine method for the used text documents.

Açıkalın and Beyazıt (2016), have used abstracts from journals and conferences as the dataset in their work and one of the datasets consists of 18 classes and the others have 34 classes. They first applied a feature selection process and have used the Latent Dirichlet Allocation method in the feature selection process. They have used Zemberek and fixed prefix 5 length as the stemming method. Naïve Bayes, Support Vector Machine and Random Forest classifiers were applied as the classifiers. The most successful result of the study is 74.90% by using SVM as a classifier. This result is received by using Zemberek. We used Naïve Bayes Multinomial as classifier in our study. The most successful F-measure

value we have achieved for the Fixed Prefix7 study; is 97.6% by the *tf* term weighting method.

Parlar and Özel (2018) used QER and Chi Square methods for feature selection; and *tp, tf, tf * idf* as term weighting methods for sentiment analysis. In the study, they used 5 different datasets as Movie, Book, DVD, Electronics and Kitchen. They also used 2 classes as positive and negative in the datasets. Naïve Bayes Multinomial is applied as the classifier. They obtained more successful results with *tp* and *tf* term weighting method. In our study we obtained more successful results with *tf* and *tf * idf* term weighting methods. When using QER and Chi Square feature extraction methods, they observed that the results for *tf * idf* increased significantly. In addition, QER feature selection yielded better results than that of Chi Square. We also used the Chi Square feature selection in our studies and found that the standard deviation based feature selection yielded more successful results.

5. CONCLUSION AND FUTURE WORK

In this study, we try to determine the most effective method of feature extraction for Turkish documents. In the study, 4 different datasets are used; 1150 News, 3000 Tweet, Turkish Email and 25 Author. The class numbers and document numbers of each dataset are different from each other. The first experiment in the study is conducted in order to find the best term weighting method. Five different term weighting methods: *tf, tp, logtf, normtf* and *tf*idf,* are compared for this purpose. As the results of the experiments show, the most successful methods are *tf* and *tf*idf* weighting methods, therefore they are used in other experiments too.

The second experiment is conducted to find the effect of stopwords on classification. When stopwords are removed from documents, it is observed that there is little or no effect on the positive direction, and therefore stopwords are removed from the datasets to reduce dataset sizes. Both experiments are conducted using the raw forms of the terms.

Zemberek, Affix Stripping, Fixed Prefix 3, 5, 7 root finding algorithms are applied to the datasets in order to search the effect of stemming methods on classification. When the results are evaluated, it is observed that the classification of the documents using the raw forms of the terms gives more successful results. Since the F-Measure values are higher when no stemming is done, all of the subsequent experiments are done using the raw forms of the terms.

We use word n-grams (1-2grams and 1-2-3grams) for feature extraction. It has been observed that n-grams (1-2grams and 1-2-3grams) increase the classification accuracy results in the positive direction however, feature set size increases. The studies are continued with feature selection procedures and our proposed standard deviation-based method, information gain, and chi square methods used in the literature are applied to reduce feature space. In the experiment of feature selection with our proposed method, the documents are classified by

using terms with a standard deviation greater than zero for each dataset and within the top 0.1, 0.5, 1, 5, 10, 15, 20 percent ranks. Standard deviation-based feature selection for different datasets have showed successful results.

Other feature selection methods that are applied are information gain and chi square methods. In this study, the terms whose information gain and chi square score values are greater than zero are taken, and the raw forms of the selected terms are used as features. When the classification F-measure results are evaluated, it is observed that standard deviation-based feature selection has more improvement than information gain and chi square methods on classification accuracy.

In this study, effects of feature extraction, feature selection, and feature weighting are investigated on 4 different text mining problems for Turkish language which is an agglutinative language. Stopwords removal and feature selection with standard deviation-based method show successful results. Stemming decreases classification accuracy which shows that better stemmers are needed for Turkish. Future studies can be done by choosing the languages in different structures that can be evaluated whether for achieving the same success. In addition, performances of other document representations such as word2vec may be investigated.

REFERENCES

Açıkalın, B., and Beyazıt, N.G., 2016. Signal Processing and Communications Applications, Zonguldak.

Akgöbek, Ö., and Çakır, F., 2009. Veri Madenciliğinde Bir Uzman Sistem Tasarımı. Akademik Bilişim 09, 11-13 Şubat Harran Üniversitesi, Şanlıurfa, 801-806.

Akın, A.A., and Akın, M.D., 2007. Zemberek, an open source NLP framework for Turkish Languages.

Albayrak, M., 2008. EEG Sinyallerindeki Epileptiform Aktivitenin Veri Madenciliği Süreci ile Tespiti, Doktora Tezi, Sakarya Üniversitesi, Fen Bilimleri Enstitüsü. 143 Sayfa.

Alpaydin, E., 2000. Zeki Veri Madenciligi (Notlar), Bilisim 2000 Egitim Semineri Notlari, Word File.

Amasyalı, M.F., and Diri, B., 2006. Automatic Turkish Text Categorization in Terms of Author, Genre and Gender. C. Kop et al. (Eds.): NLDB 2006, LNCS 3999, pp. 221–226.

Amasyalı, M.F., Balcı, S., Varlı, E.N., and Mete, E., 2012. Türkçe Metinlerin Sınıflandırılmasında Metin Temsil Yöntemlerinin Performans Karşılaştırılması, EMO Bilimsel Dergi.

Amasyalı, M.F., and Beken, A., 2013. Türkçe Kelimelerin Anlamsal Benzerliklerinin Ölçülmesi ve Metin Sınıflandırmada Kullanılması", SIU 2009, Antalya, Turkey.

Amasyalı, M.F., and Çetin, M., 2013. Eğiticili ve Geleneksel Terim Ağırlıklandırma Yöntemleriyle Duygu Analizi. SIU, KKTC.

Can, F., Koçberber, S., Balçık, E., Kaynak, C., Öcalan, H. Ç. and Vursavaş, O.M., 2008. "Information Retrieval on Turkish Texts", Journal of the American Society for Information Science and Technology, 59, 407-421.

Cataltepe, Z., Turan, Y., and Kesgin, F., 2007. Turkish Document Classification Using Shorter Roots , SIU 2007, Eskisehir, Turkey.

Ekmekçioğlu, Ç.F., and Willet, P., 2000. Effectiveness of stemming for Turkish text retrieval. Program 34 (2).

Ergin, S., Sora Gunal, E., Yigit, H., and Aydin, R., 2012. Turkish anti-spam filtering using binary and probabilistic models, AWERProcedia Information Technology & Computer Science, 1, 1007-1012.

Eryiğit, G., and Adalı, E., 2004. An Affix Striping Morphological Analyzer For Turkish, International Conference Artificial Intelligence and Applications, Austria, pp. 299-304.

Güran, A., Akyokuş, S., Bayazıt, N.G., and Gürbüz, M.Z., 2009. Turkish Text Categorization Using N-Gram Words. International Symposium on Innovations in Intelligent Systems and Applications, Trabzon, Turkey.

Han J., Kamber M., Pei J., P., 2012. Data Mining Concepts and Techniques, Elsevier ,740Pp.

Hand, D., Mannila, H., and Smyth, P., 2001. Principles of Data Mining. The MIT Press. England. 546 Pp.

İlhan, S., Duru, N., Karagöz, Ş., and Sağır, M., 2008. Metin Madenciliği ile Soru Cevaplama Sistemi, ELECO-2008, pp. 356-359.

İnan, O., 2003. Öğrenci işleri veri tabanı üzerinde veri madenciliği uygulamaları, Yüksek Lisans Tezi, Selçuk Üniversitesi, Fen Bilimleri Enstitüsü. 81 Sayfa.

Kaya, M., and Özel, S.A., 2014. Türkçe Dokümanlardaki Benzerliklerin Tespiti İçin Mevcut Yazılımların Karşılaştırılması ve Türkçe Karakter Kullanımı ile Kök Almanın Etkisinin İncelenmesi, Çukurova Üniversitesi Mühendislik Mimarlık Fakültesi Dergisi, 29(2):115-129.

Knight, K., 1991. Artificial Intelligence, McGraw Hill Inc., Second Edition, New York. 621 Pp.

Leung, K. M., 2007. Naive Bayesian Classifier. Polytechnic University Department of Computer Science / Finance and Risk Engineering. Lecture Notes.

McCallum, A. and Nigam, K., 1998. A Comparison of Event Models for Naive Bayes Text Classification. *Proceedings in Workshop on Learning for Text Categorization*, AAAI'98, 41-48.

Parlar T., Özel S.A., 2018. An Investigation of Term Weighting and Feature Selection Methods for Sentiment Analysis, Majlesi Journal of Electrical Engineering, Vol. 12, No. 2.

Patil T., Sherekar S.S., 2013. Performance Analysis of Naive Bayes and J48 Classification Algorithm for Data Classification, International Journal Of Computer Science And Applications, Vol. 6, No. 2.

Sever, H., and Bitirim, Y., 2003. Findstem: Analysis and Evaluation of a Turkish Stemming Algorithm, String Processing and Information Retrieval, pp. 238-251.

Torunoğlu, D., Çakırman, E., Ganiz, M., Akyokuş, S., and Gürbüz, Z., 2011. Analysis of Preprocessing Methods on Text Classification of Turkish Texts, International Symposium on Innovations in Intelligent Systems and Applications (INISTA), pp. 112-117, İstanbul.

Tunalı, V., and Bilgin, T.T, 2012a. Türkçe Metinlerin Kümelenmesinde Farklı Kök Bulma Yöntemlerinin Etkisinin Araştırılması, Elektrik Ve Bilgisayar Mühendisliği Sempozyumu (ELECO).

Tunalı, V., and Bilgin, T.T., 2012b. PRETO: A High-Performance Text Mining Tool for Preprocessing Turkish Texts, International Conference on Computer Systems and Technologies.

Uysal, K.U., and Günal, S., 2013. The impact of preprocessing on text classification, Information Processing and Management, pp. 104-112.

Witten, I.H., and Frank E., 2005. Data Mining Practical Machine Learning Tools and Techniques Second Edition, Elsevier, San Francisco CA, 548 pp.

Yıldız, H.K., Gençtav M., Usta N., Diri B., and Amasyalı M.F., 2007. Metin Sınıflandırmada Yeni Özellik Çıkarımı, Signal Processing and Communications Applications.

http://www.kemik.yildiz.edu.tr

CURRICULUM VITAE

Özge AKDOĞAN was born in Nizip, in 1984. She has completed her elementary educati on at Nizip Cumhuriyet Primary Education School . She went to high school at Nizip Hasan Çapan Anatolian High School . She has completed university education at department of Computer Engineering of European University of Lefke in 2008. Since 2011, she has been working as a lecturer at Nizip Vocational Scholl of Gaziantep University in Gaziantep.

APPENDIX

Appendix 1: Comparison of our study with the literature

Study	Stemming Method	Stopword Removal	Term Weighting Method	Clustering/ Classification	Feature Selection	Dataset Used
Our Study	Zemberek, Affix, Fixed Prefix 3,5,7	Applied	tf, normtf, tp, logtf, tf*idf	Classification (NBM)	Applied (1-2 grams, 1-2-3 grams, Standard Deviation, Information Gain, Chi Square)	1150 News, 3000 Tweet, Turkish Email, 25 Yazar
Ekmekçioğlu and Willet (2000)	PC-KIMMO	Not applied	Probabilistic Model	Not applied	Not applied	Turkish Papers
Sever and Bitirim (2003)	FINDSTEM, A-F and L-M algorithm	Not applied	Not applied	Not applied	Not applied	Turkish based documents
Amasyalı and Diri (2006)	Bi-gram and trigrams	Not applied	Not applied	Classification (NB, SVM, C.45, ROS, RF)	Applied (Correlation-based Feature Selection)	Turkish newspapers
Yıldız et. al (2007)	Zemberek	Applied	Trie tree, logtf*idf	Classification (NB, SVM, C.45, KNN, RF)	Not applied	Hurriyet, Vatan, Sabah
Çataltepe et al. (2008)	Zemberek	Applied	tf*idf	Classification, (Centroid)	Not applied	Milliyet, Vikipedi

Güran et al. (2011)	Zemberek, Fixed Prefix 3,5,7	Applied	Binary weighting, tf	Classification (NB, NBM, SVM, KNN)	Not applied	1150 News, Milliyet_9c_1k, Hurriyet_6c_1k
Tunalı and Bilgin (2012)	Zemberek, Affix, Fixed Prefix 3,5,7	Applied	tf*idf	Clustering	Not applied	Milliyet and NTV
Amasyalı et al. (2012)	Zemberek, 2-3 grams	Applied	tf*idf, logtf, normalize 1, normalize 2	Classification (KNN, C.45, SVM, NB, RF) Clustering (K-Means, SOM, HC)	Not applied	Ruh Hali, Film yorumları, Köşe Yazarı, Cinsiyet, Haberler, Şiir
Uysal and Serkan (2013)	Zemberek, Fixed Prefix, Porter's algorithm	Applied	Not applied	Classification (SVM)	Applied (Chi Square)	Turkish Email, English Email
Açıkalın and Beyazıt (2016)	Zemberek, Fixed Prefix 5	Not applied	Hidden Dirichlet Appointment	Classification (NB, SVM, RF)	Not applied	Paper abstracts, Conferences
Parlar and Özel (2018)	Not Applied	Not applied	Tp, tf, tf*idf	Classification (NBM)	Applied (Chi Square, QER)	Movie, Book, DVD, Electronics, Kitchen

Appendix 2: List of Stopwords

1	Acaba	54	Doksan	107	Kimden	160	Sadece
2	Altı	55	Dokuz	108	Kime	161	Sanki
3	Altmış	56	Dolayı	109	Kimi	162	Sekiz
4	Ama	57	Dolayısıyla	110	Kimse	163	Seksen
5	Ancak	58	Dört	111	Kırk	164	Sen
6	Arada	59	Edecek	112	Mı	165	Senden
7	Aslında	60	Eden	113	Mi	166	Seni
8	Ayrıca	61	Ederek	114	Milyar	167	Senin
9	Bana	62	Edilecek	115	Milyon	168	Siz
10	Bazı	63	Ediliyor	116	Mu	169	Sizden
11	Belki	64	Edilmesi	117	Mü	170	Sizi
12	Ben	65	Ediyor	118	Nasıl	171	Sizin
13	Benden	66	Eğer	119	Ne	172	Şey
14	Beni	67	Elli	120	Neden	173	Şeyden
15	Benim	68	En	121	Nedenle	174	Şeyi
16	Beri	69	Etmesi	122	Nerde	175	Şeyler
17	Beş	70	Etti	123	Nerede	176	Şöyle
18	Bile	71	Ettiği	124	Nereye	177	Şu
19	Bin	72	Ettiğini	125	Niçin	178	Şuna
20	Bir	73	Gibi	126	Niye	179	Şunda
21	Birçok	74	Göre	127	O	180	Şundan
22	Biri	75	Halen	128	Olan	181	Şunları
23	Birkaç	76	Hangi	129	Olarak	182	Şunu
24	Birkez	77	Hatta	130	Oldu	183	Tarafından
25	Birşey	78	Hem	131	Olduğu	184	Trilyon
26	Birşeyi	79	Henüz	132	Olduğunu	185	Tüm
27	Biz	80	Hep	133	Olduklarını	186	Üç

28	Bize	81	Hepsi	134	Olmadı	187	Üzere
29	Bizden	82	Her	135	Olmadığı	188	Var
30	Bizi	83	Herhangi	136	Olmak	189	Vardı
31	Bizim	84	Herkesin	137	Olması	190	Ve
32	Böyle	85	Hiç	138	Olmayan	191	Veya
33	Böylece	86	Hiçbir	139	Olmaz	192	Ya
34	Bu	87	İçin	140	Olsa	193	Yani
35	Buna	88	Iki	141	Olsun	194	Yapacak
36	Bunda	89	Ile	142	Olup	195	Yapılan
37	Bundan	90	Ilgili	143	Olur	196	Yapılması
38	Bunlar	91	Ise	144	Olursa	197	Yapıyor
39	Bunları	92	Işte	145	Oluyor	198	Yapmak
40	Bunların	93	Itibaren	146	On	199	Yaptı
41	Bunu	94	Itibariyle	147	Ona	200	Yaptığı
42	Bunun	95	Kadar	148	Ondan	201	Yaptığını
43	Burada	96	Karşın	149	Onlar	202	Yaptıkları
44	Çok	97	Katrilyon	150	Onlardan	203	Yedi
45	Çünkü	98	Kendi	151	Onları	204	Yerine
46	Da	99	Kendilerine	152	Onların	205	Yetmiş
47	Daha	100	Kendini	153	Onu	206	Yine
48	Dahi	101	Kendisi	154	Onun	207	Yirmi
49	De	102	Kendisine	155	Otuz	208	Yoksa
50	Defa	103	Kendisini	156	Oysa	209	Yüz
51	Değil	104	Kez	157	Öyle	210	Yüzden
52	Diğer	105	Ki	158	Pek	211	Zaten
53	Diye	106	Kim	159	Rağmen	212	Zira