**ANKARA YILDIRIM BEYAZIT UNIVERSITY**

**GRADUATE SCHOOL OF NATURAL AND APPLIED SCIENCES**


# TEXT CLASSIFICATION BASED ON ORGANIZATIONAL DATA USING MACHINE LEARNING

**M.Sc. Thesis by**

**Ahmed Enis ERKAYA**

**Department of Computer Engineering**

**December, 2019**

**ANKARA**

# TEXT CLASSIFICATION BASED ON ORGANIZATIONAL DATA USING MACHINE LEARNING

**A Thesis Submitted to**

**The Graduate School of Natural and Applied Sciences of**

**Ankara Yıldırım Beyazıt University**

**In Partial Fulfillment of the Requirements for the Degree of Master of Science in Computer Engineering, Department of Computer Engineering**

**by**

**Ahmed Enis ERKAYA**

**December, 2019**

**ANKARA**

## M.Sc. THESIS EXAMINATION RESULT FORM

We have read the thesis entitled **"TEXT CLASSIFICATION BASED ON ORGANIZATIONAL DATA USING MACHINE LEARNING"** completed by **AHMED ENİS ERKAYA** under the supervision of **ASST. PROF. DR. AHMET ERCAN TOPCU** and we certify that in our opinion it is fully adequate, in scope and in quality, as a thesis for the degree of Master of Science.

Asst. Prof. Dr. Ahmet Ercan TOPCU

Supervisor

Asst. Prof. Dr. Fahreddin Şükrü TORUN

Jury Member

Asst. Prof. Dr. Ali Osman ÇIBIKDİKEN

Jury Member

Prof. Dr. Ergün ERASLAN

Director

Graduate School of Natural and Applied Sciences

I hereby declare that, in this thesis which has been prepared in accordance with the Thesis Writing Manual of Graduate School of Natural and Applied Sciences,

- All data, information and documents are obtained in the framework of academic and ethical rules,
- All information, documents and assessments are presented in accordance with scientific ethics and morals,
- All the materials that have been utilized are fully cited and referenced,
- No change has been made on the utilized materials,
- All the works presented are original,

and in any contrary case of above statements, I accept to renounce all my legal rights.

**Date:** 2019, 26 December

**Signature:** **…….**

**Name & Surname:** Ahmed Enis ERKAYA

# ACKNOWLEDGMENTS

I would like to express my sincere gratitude to my supervisor, Assist. Prof. Dr. Ahmet Ercan TOPCU for his tremendous support and motivation during my study. His immense knowledge and precious recommendations constituted the milestones of this study. His guidance assisted me all the time of my research and while writing this thesis.

Lastly, I would like to express my gratitude to my precious wife for all her help. Also, I would like to add that my parents have had a great impact on this process, both with their support and their love.

2019, 26 December **Ahmed Enis ERKAYA**

# TEXT CLASSIFICATION BASED ON ORGANIZATIONAL DATA USING MACHINE LEARNING


## ABSTRACT

The increase in text data coming with the increase in the use of online platforms and the ease of access to this data have led to the increase in the number of studies on text classification. Text classification has had a great impact on such fields as spam mail detection, sentiment analysis and news categorization. Our concern in this study is Turkish text classification. While there are a lot of English papers related to text classification, the number of the studies on Turkish data is quite limited. In this study, the letters of request that came to an organization were used as the experimental dataset. These letters of request are labeled with classes. These classes are predefined in the internal processes of the organization from which the data is received. Because the letters of request used in the study came directly from the users, they contain a lot of misspellings. To correct these mistakes, normalization was applied on the text data. Then, the words in the corpus were transformed into their simple forms by morphological analysis. In addition, the list of stop words was prepared by looking at the most repetitive word groups, and they were removed. Lastly, in the preprocessing step, the repetitive classes in the corpus were simplified via the K-Means algorithm and the number of the classes was reduced. As a result, a more consistent and balanced dataset appeared. The letters of request were trained by Naïve Bayes, SVM, Random Forest, Logistic Regression and LSTM before and after preprocessing. Then, the performance of the algorithms before and after preprocessing was compared. It was concluded that the most efficient algorithm with regard to accuracy is LSTM. Moreover, the model of SaaS was developed for the organizations to benefit from the machine learning model and for increasing the data.



**Keywords:** Natural language processing, machine learning, text classification, LSTM, Naïve Bayes, Support Vector Machine, Random Forest, deep learning.

# MAKİNE ÖĞRENMESİ KULLANARAK KURUMSAL VERİLERE DAYALI METİN SINIFLANDIRMA

## ÖZ

Çevrim içi platformların kullanımının yaygınlaşması ile birlikte metin verilerinin artması ve bu verilere erişimin kolaylaşması metin sınıflandırma alanında yapılan çalışmaların sayısının çoğalmasına vesile olmuştur. İstenmeyen e-postaların tespiti ve duygu analizi gibi birçok alanda metin sınıflandırma tekniklerinin büyük katkısı bulunmaktadır. Bu tez kapsamında, Türkçe metin sınıflandırma üzerinde çalışılmıştır. Metin sınıflandırma ile ilgili İngilizce birçok araştırma olmasına karşın, bu alanda Türkçe veriler üzerinde yapılan çalışmalar oldukça azdır. Bu çalışmada, bir organizasyona gelen şikayet mektupları eğitim verisi olarak kullanılmıştır. Şikayet mektupları ilgili konular ile etiketli bir şekildedir. Bu konular verilerin alındığı organizasyonun kendi iç süreçlerinde tanımladığı konulardır. İncelenen şikayet mektupları direkt olarak kullanıcıdan geldiğinden dolayı çok fazla yazım yanlışı içermektedir. Bu yazım yanlışlarını düzeltmek için metin verileri üzerinde normalizasyon işlemi uygulanmıştır. Daha sonra veriler üzerinde biçimbilimsel analiz uygulanarak külliyat içerisinde bulunan kelimeler yalın hale getirilmiştir. Ayrıca en sık tekrar eden kelime gruplarına bakılarak etkisiz kelimeler listesi oluşturulmuş ve etkisiz kelime olarak görülen sözcükler temizlenmiştir. Son olarak ön işleme adımında veri kümesinde tekrar eden sınıflar K-Means algoritması ile sadeleştirilmiştir. Bu sayede daha dengeli ve mantıklı bir veri kümesi oluşturulmuştur. Gelen şikayet mektupları ön işlemeden önce ve sonra Naïve Bayes, SVM, Random Forest, Logistic Regression ve LSTM ile eğitilmiştir. Ön işleme öncesi ve sonrası algoritmaların performansları kıyaslanmıştır. Sonuç olarak, doğru tahmin etme açısından en verimli çalışan algoritmanın LSTM olduğu görülmüştür. Ayrıca kurumların geliştirilen modelden faydalanması ve verilerin arttırılması amacı ile SaaS modeli geliştirilmiştir.

**Anahtar Kelimeler:** Doğal dil işleme, makine öğrenmesi, metin sınıflandırma, LSTM, Naïve Bayes, Support Vector Machine, Random Forest, derin öğrenme.

# CONTENTS

# NOMENCLATURE

## Acronyms

| | |
|---|---|
| AI | Artificial Intelligence |
| CRM | Customer Relationship Management |
| DBMS | Database Management System |
| K-NN | K-Nearest Neighbors |
| LR | Logistic Regression |
| LSTM | Long Term-Short Memory |
| ML | Machine Learning |
| NB | Naïve Bayes |
| NER | Named Entity Recognition |
| NLP | Natural Language Processing |
| PCA | Principal Component Analysis |
| POS | Part of Speech |
| RF | Random Forest |
| RNN | Recurrent Neural Network |
| SVM | Support Vector Machine |
| TF | Term Frequency |
| IDF | Inverse Document Frequency |

## LIST OF TABLES

## LIST OF FIGURES

# CHAPTER 1

## INTRODUCTION

### 1.1 Overview

The development of machine learning techniques has brought solution proposals for our daily problems. With the development of internet, people's rate of using personal computers and mobile phones has increased. They share a lot of things like their location, photos on social media and other online communication platforms including even their personal data. People's view of daily life problems has changed a lot. They now share problems they meet in daily life in a quick and easy way on diverse social media platforms for both complaint and good feedbacks. It has become inevitable for firms with commercial concern and organizations with political concern to adapt it. Organizations and firms have begun to follow and respond to these types of feedbacks on both social media platforms and their own CRM software.

In this study, text classification was applied to the letters of request that came to an organization. The methods applied and the results acquired were examined in a detailed way. In addition, for the organizations to benefit from the machine learning model and for increasing the data, the SaaS application was developed.

### 1.2 Customer Relationship Management

People oriented organizations and firms need to listen to the feedbacks coming from people about their services or products and to direct themselves according to their reviews. Organizations use CRM systems to analyze their interaction with people and to keep the statistical data. CRM systems have a database keeping the personal information of people benefiting from a service or product in a detailed way. In addition, the feedbacks and the requests coming are saved in this database. With the development of technology, today, these feedbacks and requests are conveyed to organizations via variable communication platforms and social media along with e-

mail and phones. CRM systems work integrally with these communication platforms and transfer the requests to their database no matter which platform they come from.

We meet such types of CRM as Contact Center CRM, Social CRM, Mobile CRM and Business-to-Business CRM in practice. In CRM systems used as a contact center, the personnel in charge of the communication center conveys the requests to the department of marketing. Who is in contact with the client is the call center personnel. Social CRM works on the messages sent to the organization via social media. The client is in direct contact with CRM. Mobile CRM works as a bridge between the client and the organization via mobile applications developed for smart phones and tablets. It can achieve the location of the users by using their smart phones. Business-to-business CRM helps the issues be addressed to just in the beginning step [1]. In the organization which the data used in the study was taken from, a CRM system integrating all these types of CRM is used.

## 1.3 The Aim of The Study

Organizations offer diverse online application forms for people to convey their complaints to related departments. Hundreds of complaint or content letters are conveyed to big organizations via CRM software every day. It is also very important for these letters to be followed and responded to for people's content. Time is the most important factor in the process of responding. People expressing their requests expect them to be fulfilled as soon as possible. In big organizations, it causes waste of time to convey these requests coming via CRM software to the related department/staff, and people's rate of content decreases [2]. The misclassification of the incoming letter of request causes the time spent for responding to be longer. The loop indicated with red arrows in Figure 1.1 is a result of misclassification, which leads to waste of time.

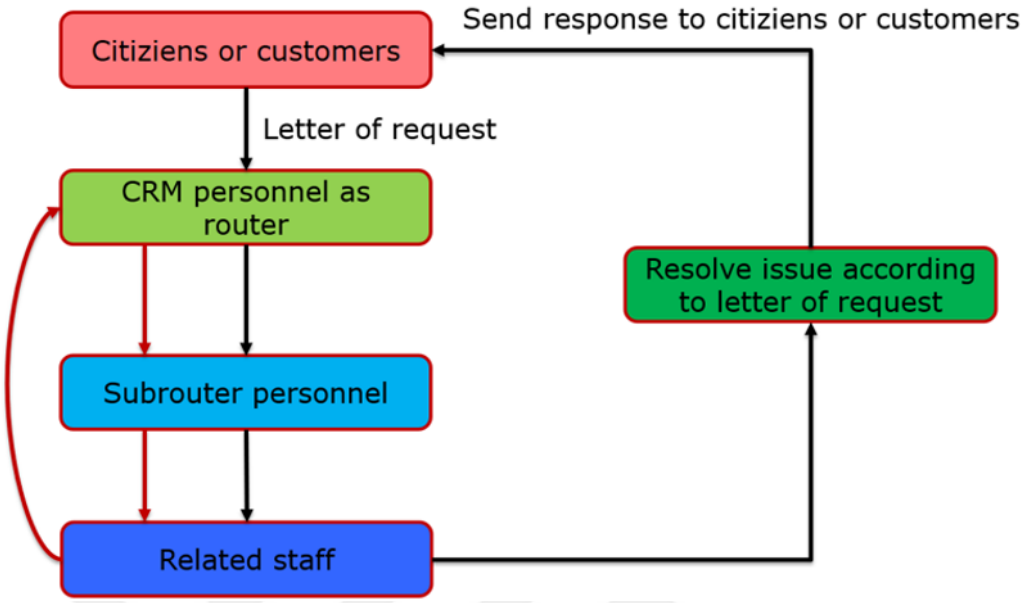


**Figure 1.1** Process of responding

It is the main aim of this study to detect the class of the letters of request by applying diverse learning techniques on them and to respond to them in a faster way. Via classification algorithms, it is expected for the letters to be conveyed to the related class in a fast way, and thus, for the staff to respond to the letters in a faster way. Which classification algorithm is better for this process is another point to detect. There is also a challenge for this classification process in this study, because all the letters of requests, which were taken from a Turkish organization, are in Turkish.

## 1.4 Thesis Organization

In the first chapter of this study, which consists of five main chapters, the content of the study and its aim and gainings are given. Also, CRM system and its types are mentioned. In the second chapter, similar studies are examined under the title of literature review. In the third chapter, the algorithms, libraries and technology used in this study are explained in a detailed way, and detailed information about the data are given. In the forth chapter, the results having been acquired with the working of the algorithms and the results are evaluated. The last chapter incorporates the conclusion and future works parts.

# CHAPTER 2

## LITERATURE REVIEW

In this study, text classification is performed by using supervised learning algorithms. The dataset to be used in the training is Turkish. In this section, the use of the algorithms and the methods having been used in the other studies were analyzed.

### 2.1 Text Classification

Pranckevicius and Marcinkevicius, in their study, analyze Naïve Bayes, Random Forest, Decision Tree, SVM and Logistic Regression, which are of the algorithms used in text classification, according to their accuracy. They also state that text classification, which is performed by the combination of machine learning and NLP techniques, is a field which has been studied on recently by many scientists. In the related study, 2638274 comments having been sent to an e-commerce application are analyzed under five classes. After the preprocessing steps, Tomas Pranckevicius and his friend argue that the accuracy of Logistic Regression (58.50%) is better than the other algorithms, and the lowest accuracy is Decision Tree's [3].

One of the algorithms which are often used to solve text classification problems is Naïve Bayes. Its qualities "easy implementation" and "high accuracy rate" have made it popular. The paper [4] tells about a study on text classification of English tweets via Naïve Bayes algorithm. In this study, whether the tweets are positive or negative was determined. In the process of text classification, the algorithm was worked after preprocessing. Such terms as url and username contained in the tweets were removed. Reduction was performed for the replicated characters.

In another text classification study, a classification process on the abstracts of some research papers was performed. In this study, some research papers belonging to four classes named DBMS, Operating Systems, Java and Data Structure were examined, and Naïve Bayes classifier, Decision Tree and their own algorithm were used. It is concluded in this study that Decision Tree algorithm reached a great accuracy, and

Naïve Bayes showed a notable performance. As in the other text classification studies, preprocessing was performed in this study, as well, before the model was not trained with classification algorithms. Porter Stemmer and a tokenizer were used to extract keywords from the abstracts [5].

Bhumika et al. in [6] did a study on text classification algorithms. The document classification processes and classification algorithms were examined in a detailed way. In this study, the tasks of text mining were divided into five categories as text categorization, text clustering, concept mining, information retrieval and information extraction. The researchers of the study define text categorization as labeling pre-defined categories and documents, text clustering as grouping similar documents, information retrieval as reading information from documents according to users' request and information extraction as responding to questions.

Korde and Mahender in [7] point out that over 80% of all the online data is placed as text and has quite a high commercial value. The researchers of the study preprocessed the data to be used for text classification for cleaning it. They state that the raw data affects the training cost not only in terms of speed but also in terms of result quality. They also analyzed the advantages and disadvantages of classifiers in a detailed way and presented them as a table. In this study, it is stated that Naïve Bayes and SVM algorithms work well with textual and numerical data and are implemented in an easier way compared to the other algorithms. It is also pointed out that Naïve Bayes works well only on independent data, which is a negative aspect of Naïve Bayes, because daily problems are generally correlated.

All the related research above shows that the datasets used for text classification are usually in English. Along with it, studies done with Arabic datasets [8-10] are also common.

In [8], a study Arabic text classification which was performed on, it is stated that the majority of online information is available as text, thus it is important to acquire sensible information from text data using text mining methods. The researchers state that in the study, along with the studies concerned with text classification in English, there are text classification studies in other European languages such as German,

Italian and Spanish and Asian languages such as Chinese and Japanese, as well. The Arabic data, which was used in this study, was taken from an online news channel broadcasting in Arabic. The Maximum Entropy classifier, which is a supervised method, was used in the study to classify the news to analyze into six categories. The data was processed before the training of the model. In preprocessing, firstly, all the data was transformed into UTF-8, then, the punctuation marks and non-letters were removed. Later on, such NLP methods as tokenization and stemming were applied, thus the data was cleaned. The datasets having been acquired without preprocessing, "with normalization", "with normalization plus tokenizer" and "with normalization plus tokenization plus part of speech methods" were compared to each other in terms of precision. As a result, it is concluded that the consistent result was taken from the dataset which was acquired with normalization, tokenizer plus POS. Parts of speech were used to label the text as nouns and proper nouns.

Mohammad et al. used SVM, Naïve Bayes and Neural Network in [11], which is another text classification study having been done in Arabic. It is pointed out in the study that preprocessing is an indispensable step to reduce the complexity of the document. This study divides preprocessing into three steps, and summarizes the first step as stop word cleaning, the second step as generating vector space models and the last step as selecting the most useful and valuable features. This study was done on 1400 Arabic documents belonging to eight categories and having been taken from diverse online news sites. 63% of the data was used for training, and the rest was used for testing. According to the test results, the researchers state that the most consistent algorithm is SVM. As for text classification with Turkish data, its examples are available in studies like [12, 15-17].

In [12], the researchers compare SVM, Naïve Bayes, C 4.5 and Random Forest algorithm to each other in terms of author name, document genre and author gender. They concluded after examining the previous studies that classification is performed according to such criteria as commonly-used words, word length and sentences length to determine the author name. In their own study, they used the character n-grams method to reach the text classification aims. The writings of eighteen authors in different topics which were collected from three different daily newspapers were used

as dataset. The dataset incorporates eighteen authors, thirty-five different texts having been written by each author, three text topics “policy”, “popular interest” and “sports”, four females and fourteen male authors. The correlation-based feature selection method was used for feature selection in this study. It was concluded that feature selection boosted the accuracy of the classification. The best working algorithm in author detection was seen to be Naïve Bayes. In genre and gender detection, it is SVM.

In [15], the researchers compare the traditional bag-of-words model to the artificial neural network document representation method. In the dataset, there is text data belonging to seven categories (economy, policy, sports etc.) and each category has seven hundred texts. In another dataset in this study, there are six different categories each with six hundred texts. In the study, such preprocessing steps as stop word cleaning and morphological analysis were applied. To implement text classification algorithms, Python programming language and genism, nltk, sklearn and libsvm libraries were used. Naïve Bayes was used as a classification algorithm. The researchers state that their reason for choosing this algorithm is the fact that it works more efficiently as compared to the others. In contrast to the studies in literature, they state that morphological analysis and stop word cleaning in their own dataset did not have an impact on the result. They also draw attention to the importance of feature selection in text classification and used the approaches of Knowledge Gain and Chi-Square.

In [16], the researchers state that Turkish is more limited as compared to the other languages, thus there are not many text classification studies in Turkish. In their study, they applied Bayesian Probabilistic classifier, Nearest Neighbor classifier and Decision Tree algorithms with different datasets represented with uni-gram, bi-gram and tri-gram words. The studies were performed on six hundred text documents having been collected online and belonging to the categories “auto”, ”policy”, ”medicine”, ”magazine”, ”economy” and “sports”. Each category has one hundred documents. The researchers represented the text documents via TF-IDF. In their study, in which they state that millions of words can derive from a word root, they applied stemming word to their data to be able to reduce the vector space. In preprocessing, respectively, they transformed the data to lowercases, then made a stop word list and applied stop word

cleaning. Later on, they performed stemming word, and performed feature selection with the information gain method**.** The algorithms were worked by preprocessed data. As a result, K-NN algorithm was seen to be the worst one. The researchers also state that the wideness of the feature space affects the result in a negative way. To conclude, the narrower the feature space is, the better the algorithms work.

In [17], the impact of preprocessing in e-mail and news data in Turkish and English on text classification was examined in a detailed way. The researchers of the study state that text classification is performed by the steps of classification, feature extraction, feature selection and classification, and the steps of preprocessing are tokenization, stop-word removal, lowercase conversion and stemming. It is also stated in the study that English is an example of non-agglutinative languages while Turkish is an example of agglutinative languages commonly used in the world. In other words, it is seen that English and Turkish are suitable for comparison in text classification. In this study, SVM was used as a text classification algorithm, and Micro-F1 was used for measuring. According to the result, they state that the preprocessing step is as important as the steps of feature extraction, feature selection and classification. In addition, while stop-word cleaning was taken as an unimportant detail compared to the other studies, this study shows us that it is quite an important step. It is also argued in this study that some preprocessing tasks like lowercase transformation are indispensable regardless of the domain and the language, but it is necessary for some other tasks to be combined according to the domain and the language.

In [57], Yangsen et al. mention their research on sentiment analysis for text. The major challenge in sentiment classification is how to determine the key part of comprehending the sentiment in text. The researchers of this study proposed the model “Coordinated CNN-LSTM-Attention (CCLA)” to find a solution for this problem. In this process, they used the advantages of CNN and LSTM to apply vector representation on the sentences. As a result, they presented their CCLA architecture. They stated in their paper that they preferred CNN because of its ability of capturing semantic information from n-grams. In the model of sentiment classification, they used the softmax function. They also used the dataset of movie review including 5331 positive and 5331 negative reviews in their experiment. Most of these reviews consist

of just one sentence. For document-level sentiment analysis, they used IMDB dataset. It consists of 100000 moview reviews, and is divided into three parts: 25000 labeled train instances, 25000 labeled test instances and 50000 unlabeled instances. Each review can be positive or negative. In the subjectivity classification task, the researchers used a subjectivity dataset consisting of 5000 subjective and 5000 objective instances. It is concluded that their proposed method CCLA showed the best performance with regard to accuracy in their experiment on these datasets.

Guimarães et al. in [58] mention their research on prediction of users' age based on social network messages via deep learning. They argue that it will be easier to do sentiment analysis from social media messages if users' age is known. Their research suggests that social media users who are the same age share text about the same topics. The researchers used a dataset consisting of 7000 sentences with three age groups (teenager, adult, general). They concluded that Convolutional Neural Network showed the best performance with 95% precision in their study in which different machine learning algorithms were used. Firstly, before training the text data with the algorithms, they removed the punctuation marks. They took the emoticons used in social networks as punctuation marks. They also concluded that teenagers extend the spellings of the words they commonly use in the step of collecting data. In addition, while teenagers use fewer characters per word in a sentence, adults use more characters. Then, the data was trained with Decision Tree, Random Forest, SVM, CNN and Multilayer Perceptron, and the highest precision value was achieved with CNN. What is more, with the trained model, the researchers presented an age prediction application.

Zhang et al. in [59] state that they applied the method of character level classification in addition to the traditional methods. The model of bag-of-words and the models based on it such as TF-IDF focus on the words in sentences. The model of bag-of-words deals with which word is contained in text. As for TF-IDF, it determines the importance of the words and the stop words. Then, the bag of n-grams present structural information about the text data. All of these methods are seen to be based on the word. The algorithm of RNN performs the classification detecting the relation between these words. Zhang and his friends, who examined the studies having been

done with Convolutional Networks, concluded that CNN is useful for extracting information from raw signals, ranging from computer vision applications to speech recognition and others. In their study, they took the characters as raw signals. They are who gave the first example of applying ConvNets only on characters. They used eight different datasets containing 120000 to 360000 train samples and 2 to 14 classes. These datasets are AG's News, Sogou News, Amazon Review etc. The researchers point out in their study that the model of character-level ConvNet is the most effective method.

Parwez et al. in [60] mention their study of multi-label text classification on microblogs using CNN. They point out that microblogging sites include a lot of text data, and a lot of applications such as information filtering, user profiling, topical analysis and content tagging are available today. It is also stated in their study that traditional machine learning algorithms usually use the techniques of bag of words or n-gram to generate feature vectors, but the microblogging text is not big enough for these techniques. The researchers did a comparative analysis between the proposed CNN models, traditional machine learning models and one of the existing CNN architectures. Then, they built their own models combining diverse CNN models. Parwez at al. performed their experiment on the disease-related tweets consisting of 4320 training instances and 480 validation instances and four different classes as dengue, malaria, influenza and cholera. Then, they gave the suitable parameters such as word embedding, whose size is given as 200 in the study, and dropout parameter at embedding layer, whose probability is given as 0.15, to be protected from overfitting. They also argue that their proposed CNN models will be useful for tweet streams or text streams generated by different online social media platforms.

## 2.2 Natural Language Processing

Natural Language Processing is a subfield of AI dealing with human language processing in a way that the computer can understand. NLP, which was developed for computers to understand the language humans speak to each other, is becoming more and more popular. The fact that it has become easier to access people's speech with the help of both articles written via social media and the increase of communication channels like television and radio has expanded the working areas of NLP. In text

classification, NLP is needed for analyzing the data coming from users. It is quite a hard step to make a process by making sense of the text data. The research [18] shows that NLP is taken under four main steps. In lexical analysis, sentences are split into smaller parts called tokens. In the step of syntactic analysis, the sentence structure (syntax) of all the tokens are taken. Whether their syntax is regular or not is checked. In semantic analysis, the meanings of sentences are interpreted according to the previously steps. The data is transformed into output data and the steps of NLP are finished.

In this study, a morphological analyser, which is an important component of NLP, was used in preprocessing. Zemberek library was used for Turkish morphological analysis.

Zemberek uses root dictionary-based parses to perform parsing. The library performs parsing by finding the probabilities of the root. The library firstly reads the binary root file contained inside. When the root of the word is read, the related statements are added to the root. As for the result, it is stored as Direct Acyclic Word Graph (DAWG) tree (see Figure 2.1). In Turkish words, the root of the word might be in different forms. For these statements, different probabilities have been added to the tree, as well, for the words deformed. For example, the Turkish word “kitaba” does not contain the word “kitap”, which is the root of the word. That is why, the word “kitab”, which is deformed, also has been added to the tree [19] (see Figure 2.2).

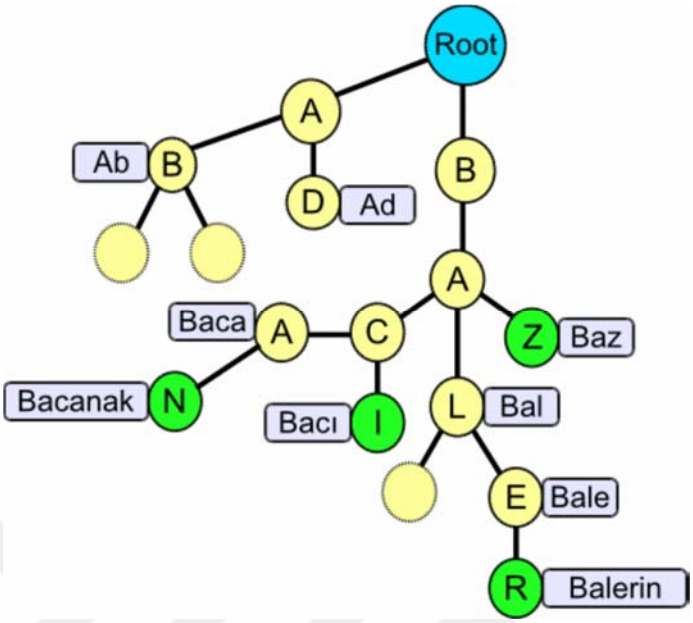


**Figure 2.1** Tree structure [19]

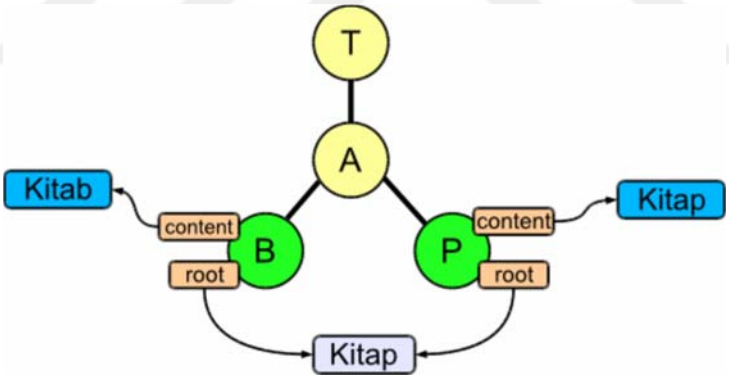


**Figure 2.2** Tree structure with a deformed word [19]

Another method used for Turkish morphological analysis is TRMOR. Kayabas et al. in [20] present a Turkish morphological analyser using SFST tool (Stuttgart Finite-State Transducer). The performance of TRMOR was tested by one thousand words having been selected randomly from wikipedia. TRMOR reached 94.12% precision value. TRMOR firstly connects stems to suffix morphemes evaluating all the propabilities and then maps the result string in the correct surface form via morphological rules. Because Turkish is an agglutinative language, it is quite hard to parse compound words. In this study, the example of “acemborusu” is given for this

statement. While “su” is the compound marker in the word “acemborusu”, “i” is the compound marker in the word “ayçiçeği”. In the study, it is pointed out that every compound word cannot be handled now, but this handling might be possible in future works.

Oflazer in [21] tells about his research on Turkish NLP. Oflazer, who states that Turkish is an agglutinative language in his paper, points out that Turkish is so hard a language for NLP with regard to morphological analysis. The Turkish word roots have been affected by Arabic, Persian, Greek, Armenian, French, Italian, Germany and recently English. Geographical, culturel, commercial and temporal proximity is the reason for this interaction. The words in one sentence can take many inflectional and derivational suffixes. A word in Turkish itself can be almost a sentence in English as in the example below.

yap+abil+ecek+se+k => if we will be able to do (it)

In his paper, in addition, he also states eight studies having been done in the past on Turkish NLP. Although Turkish is a language used by more than sixty million people in a wide geographical area, it is pointed out that the studies on this area have started in the last two decades. Despite all the challenges, the last several years have seen significant increase of research, he states.

Dai et al. in [9] propose a novel for relation classification, which plays an important role in acquiring sensible information from unstructured text. Relation classification plays an important role in natural language processing applications such as information extraction, knowledge base completion and question answering. Taking the sentence “[Shakespeare_e1] was born in [Stratford_e2]” into consideration, the words “Shakespeare” and “Stratford” are taken as two different entities. Relation classification systems aim to automatically select suitable relationships like the place of birth between e1 and e2 in a predefined relation collection. In the traditional approach, handcrafted features including lexical, semantic or syntactic information from original text have been used to acquire a better classification performance. In their paper, the researchers state that Deep Neural Networks (DNN) have been used recently. Firstly, the input representation layer presents the word and part-of-speech

tags representation. Then, bidirectional recurrent neural network and tree structure recurrent neural network were used. Subsequently, such representations produced by sequence LSTM and tree structure LSTM are fed into a full-connected layer. Lastly, a softmax layer is used to predict the relation between entities. Then, they test the method with the datasets of Wikipedia and SemEval. As a result, they conclude that their proposed method works better than the existing approaches.

# CHAPTER 3

## RESEARCH METHODOLOGY

In this section, the dataset used to be classified in the study is explained in a detailed way. Both the structure of the words and the number of the uni-grams, bi-grams and tri-grams included in the dataset both before and after the preprocessing steps are also stated in this section. Another point stated is the algorithms used in the study and the technologies used to implement them. The software language and related technologies used to develop the SaaS application are also given in this section.

### 3.1 Architecture Design

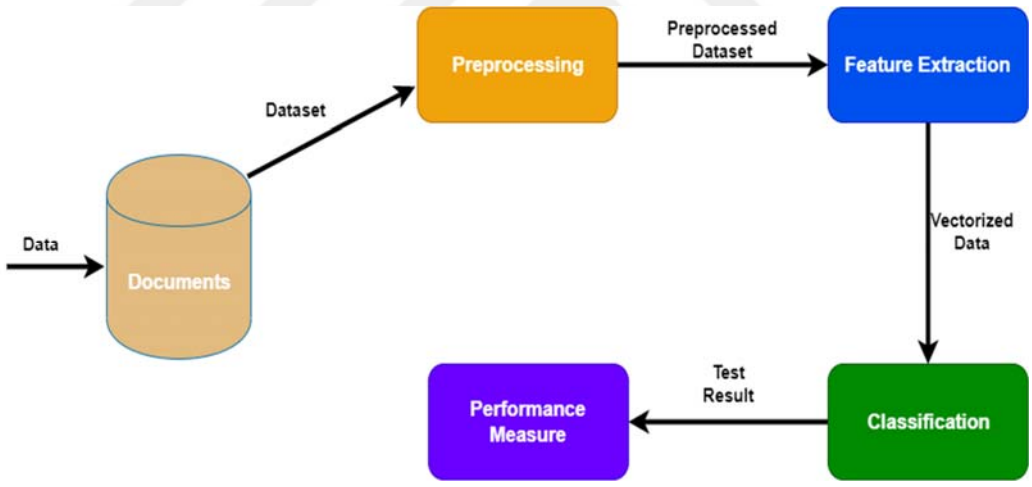


**Figure 3.1** Classification arhitecture

There are a lot of different approaches to perform classification. Figure 3.1 shows the methods used in this study. Because supervised learning techniques are applied, we need to have labeled documents to start performing text classification. After preprocessing, the data takes some processes for the algorithm to work better. After applying classification algorithms, the results are compared with one another in terms of accuracy and training time.

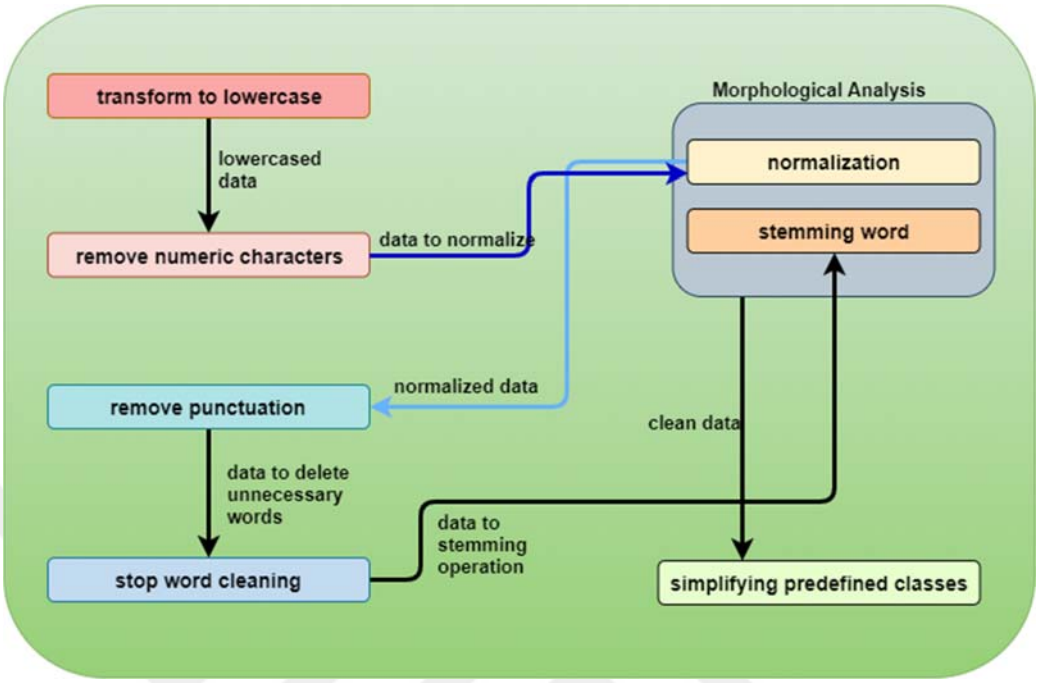


**Figure 3.2** Preprocessing steps

In the step of preprocessing, firstly, all the documents in the corpus were transformed into lowercases. Secondly, the numerical characters were removed, then normalization as a morphological analysis step, and removal of punctuation, stop word cleaning and the stemming word step were performed, and the predefined classes were simplified (see Figure 3.2). Lastly, the data was worked to extract feature vectors via the TF-IDF algorithm. The data vectorized via the TF-IDF algorithm was trained by Naïve Bayes, SVM, Random Forest, LSTM and Logistic Regression respectively, and lastly, the results were compared to each other.

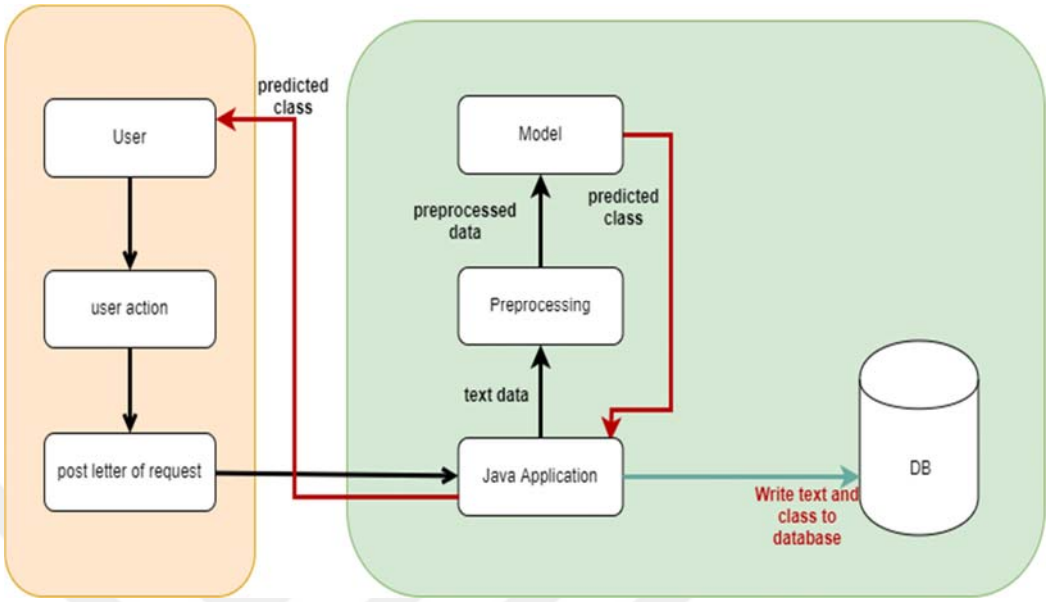


**Figure 3.3** The SaaS model

More data is needed to improve the model built after training. A web application was developed both for increasing the number of the text datas and for the organizations to be able to use the model having been built in this study. The user conveys the letter of request whose class he/she wants to be determined to the system via the software interface. There is a Java application in the system welcoming the request coming from users. The application conveys the letter of request to the module of preprocessing. The preprocessed text data is conveyed to the machine learning model for prediction of its class. The model makes prediction and conveys the response via the Java application to the user. The letter of request and the predicted class are written into the database. In this study, React.js for the frontend, Java Spring Boot for the web application and PostgreSQL for the database were used.

## 3.2 Data Exploration

The data used in this study was taken from an organization hosting hundreds of letters of request in a day in Turkey. The dataset consists of 225239 letters of request and 1819 topics.

**Table 3.1** Comparing the number of the words

| Dataset | Min. word count | Max.word count | Average word count |
|---|---|---|---|
| Raw | 1 | 3026 | 33 |
| Preprocessed | 1 | 1171 | 18 |

Table 3.1 presents the comparison of the number of the words contained in the raw dataset and the preprocessed dataset. According to the table, before preprocessing, a letter of request consists of approximately 33 words. Then, the letters of request consist of at least 1, at most 3026 words. Table 3.1 also gives us information about the number of the words in the documents contained in the corpus after preprocessing. A document consists of approximately 18 words. It is also stated that the documents consist of at least 1, at most 1171 words. It is seen that approximately 15 words were removed from each letter of request via preprocessing.

In addition, when the un-processed data is examined, the most repetitive 20 uni-grams are shown in Figure 3.4 and the most repetitive bi-grams are shown in Figure 3.5. It is seen that a lot of unnecessary words are repeated often in the corpus when the most repetitive words and word groups are looked at. For instance; the spelling “ni” is repeated more than 200000 times. In fact, this spelling itself has no meaning in Turkish. The misspellings of the users cause such types of spellings to be included in the letters.

When we look at the words repeated, we see that words like “gereği”, “arz” and “talep” are used often, because the data in this study was taken from the letters of request of an organization. In this study, the structures including words and word groups like “Gereğini arz ederim”, ”Yapılmasını talep ederim.”, “Sayın” and ”başkan” were taken as stop words, and they were all cleaned in the process of preprocessing. In Figure 3.4, which tells about the word cloud before preprocessing, the word groups including the stop words mentioned before are easily seen. In addition, the conjunction “ve” in Turkish is one of the repeated words. Along with the stop words predefined for Turkish [22], our own custom stop words are also defined (see Appendix B).

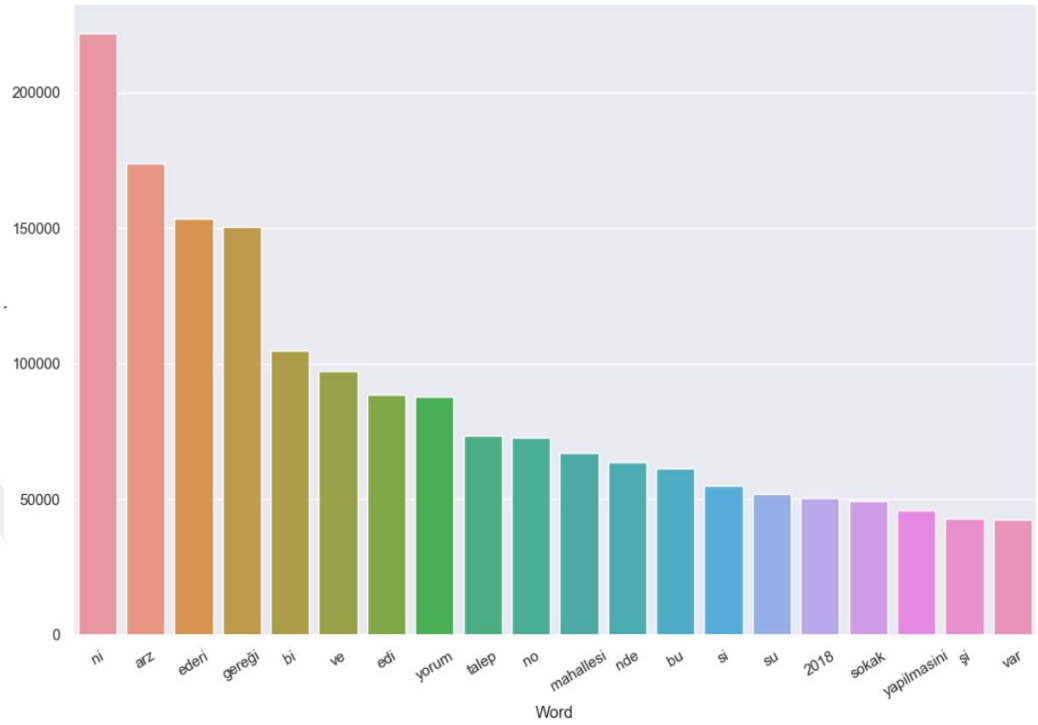


**Figure 3.4** Most frequently occurring uni-grams before preprocessing

Along with the unnecessary words and misspelled words, the fact that “2018” is among the most repetitive uni-grams shows us that it is needed to remove the numerical characters via preprocessing, as well.

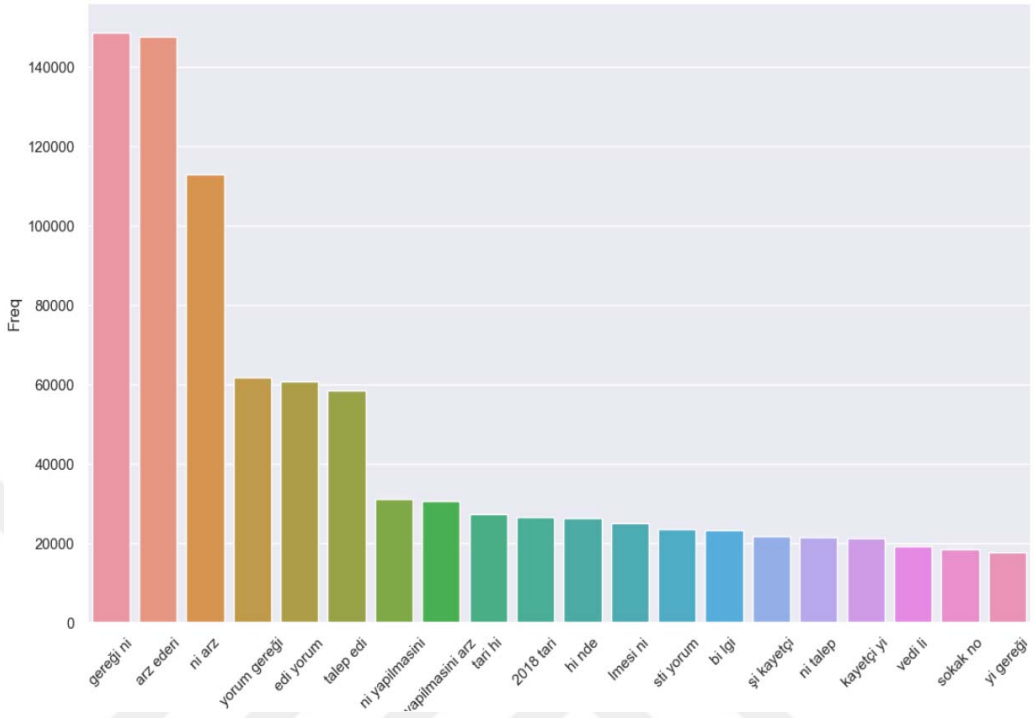


**Figure 3.5** Most frequently occurring bi-grams before preprocessing

In Figure 3.5, it is seen that the most repetitive bi-grams are bi-grams like “sokak no”, used by the citizens to give their address, and “şi kayetçi” and “ni talep”, used by them to make their requests, along with the stop words like “gereği ni” and “arz ederi”. In addition, because of the misspellings of the citizens, some of the words making the bi-grams are seen to be meaningless.


**Figure 3.6** Word cloud before preprocessing

The above Figure 3.6 gives us a summary about the data before preprocessing. Word clouds or tag or text clouds are figures in which the words in the corpus are written in a bigger size and bold according to their rate of repetition, and the other words are written in small font. The word cloud in Figure 3.6 is seen to be in harmony with the most repetitive words in Figure 3.4 and Figure 3.5.

Taking the state of the data in the corpus before preprocessing into account, on the data, the steps of;

- Transformation to lowercase
- Removal of numerical characters
- Correcting misspellings via normalization
- Removal of punctuation
- Stop word cleaning
- Stemming word

  were performed respectively and a cleaner and understandable dataset was acquired.

The steps of preprocessing are also given in our architecture design in Figure 3.2. It is possible to see a sample letter of request examined by all these preprocessing steps in

Figure 3.7 below. The numerical characters and the stop words were removed as is seen in the figure. Lastly, morphological analysis was performed on the letter of request.

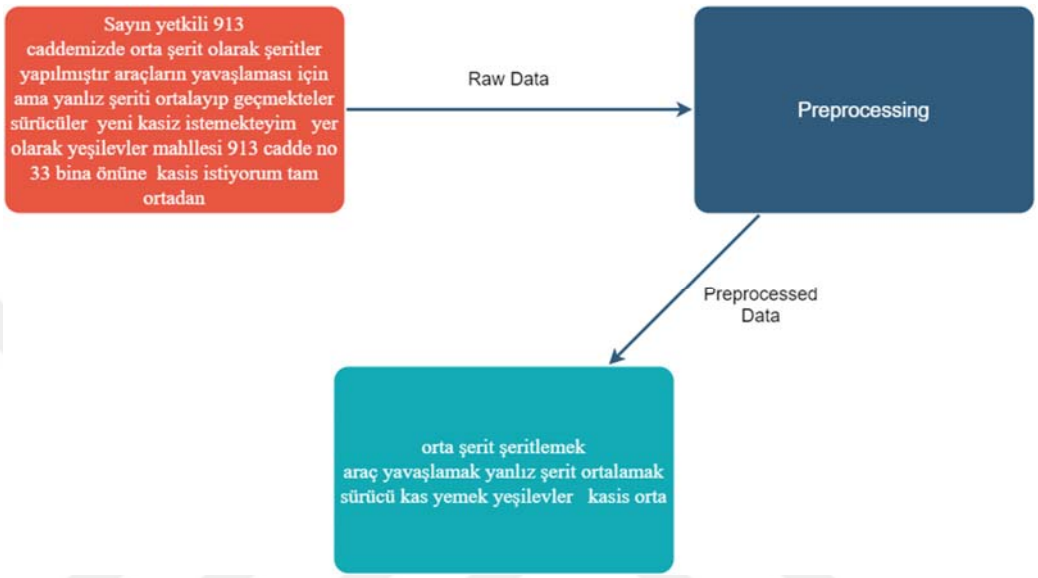


**Figure 3.7** Illustration of preprocessing

The most repetitive twenty uni-grams after preprocessing are shown in Figure 3.8, and the most repetitive twenty bi-grams are shown in Figure 3.9. Among the most repetitive uni-grams and bi-grams after preprocessing, sensible keywords like “su”, “durak” and “çalışmak” are seen.

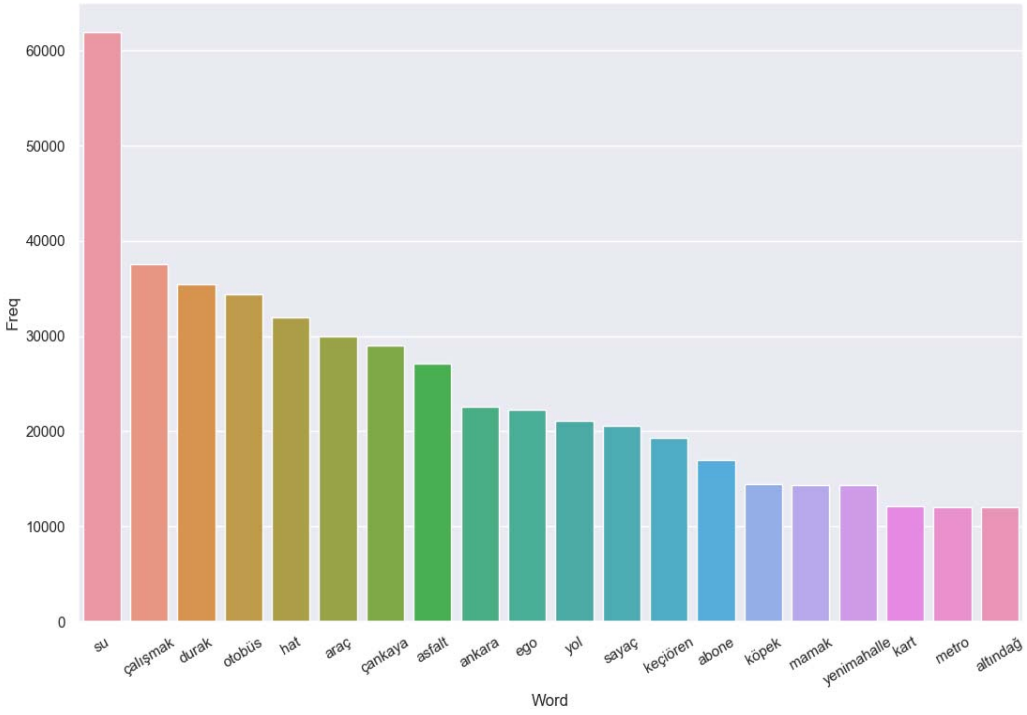


**Figure 3.8** Most frequently occurring uni-grams after preprocessing

It is seen that the keywords acquired after preprocessing are more consistent when looking at the most repetitive uni-grams in Figure 3.8. The keywords like “su”, “çalışmak”, “durak” and “hat” are among the working fields of the organization and it will help detecting the related classes. Comparing Figure 3.8 with Figure 3.4, the uni-grams acquired after preprocessing are seen to be more consistent and understandable.

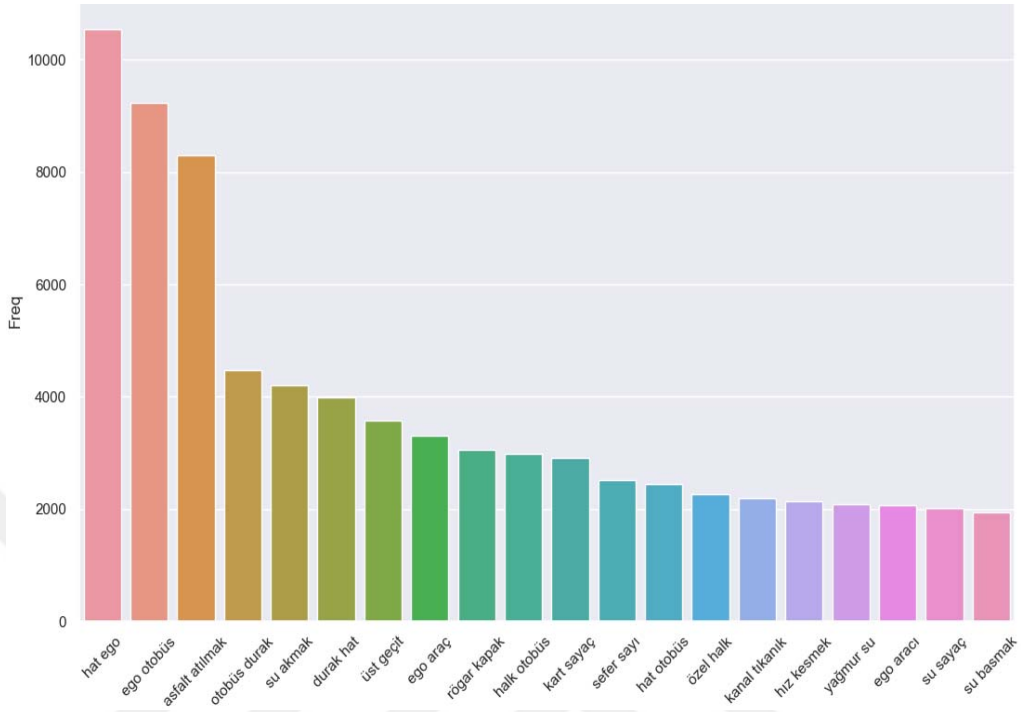


**Figure 3.9** Most frequently occurring bi-grams after preprocessing

In Figure 3.9, the most repetitive bi-grams are presented. In the figure, the keywords of the classes which are probable to be related to each other and the classes like EGO and ASKI, which are used as labels in our dataset, are seen. It can be easily said the data has become more sensible via preprocessing.


**Figure 3.10** Word cloud after preprocessing

In the word cloud in Figure 3.10, the fields the organization deals with are easily seen. Preprocessing will improve the classification performance.

**Table 3.2** Uni-gram frequency comparison according to the preprocessed data

| uni-grams before preprocessing | | uni-grams after preprocessing | |
|---|---|---|---|
| ARZ | 156978 | su | 61890 |
| EDERİM. | 101393 | çalışmak | 37593 |
| GEREĞİNİ | 94992 | durak | 35481 |
| TALEP | 64936 | otobüs | 34370 |
| MAHALLESİ | 54859 | hat | 32008 |
| ve | 53204 | araç | 29953 |
| YAPILMASINI | 45722 | çankaya | 29052 |
| EDERİM | 44378 | asfalt | 27101 |
| VE | 43057 | ankara | 22500 |
| SOKAK | 39322 | ego | 22281 |
| SU | 38231 | yol | 21152 |
| bir | 20609 | sayaç | 20609 |
| CADDESİ | 30665 | keçiören | 19309 |
| bu | 26527 | abone | 17007 |
| EDİYORUM. | 26245 | köpek | 14453 |
| TARİHİNDE | 26240 | yenimahalle | 14322 |
| GEREĞİNİN | 25847 | mamak | 14317 |
| ÖNÜNDE | 25734 | kart | 12179 |
| SAAT | 24901 | metro | 12029 |
| ÇANKAYA | 24534 | altındağ | 12025 |

In Table 3.2, the uni-grams acquired before and after preprocessing and their frequency are given. The unnecessary words taken as a stop word and whose frequency is quite high before preprocessing were removed during the preprocessing steps. It can be easily said that a serious data cleaning was performed via stop word cleaning when looking at the comparison in Table 3.2, Table 3.3 and Table 3.4.

**Table 3.3** Bi-gram frequency comparison according to the preprocessed data

| **bi-grams before processing** | | **bi-grams after preprocessing** | |
|---|---|---|---|
| gereği ni | 148561 | hat ego | 10541 |
| arz ederi | 147666 | ego otobüs | 9229 |
| ni arz | 112855 | asfalt atılmak | 8306 |
| yorum gereği | 61705 | otobüs durak | 4471 |
| edi yorum | 60737 | su akmak | 4197 |
| talep edi | 58599 | durak hat | 3990 |
| ni yapilmasini | 31116 | üst geçit | 3576 |
| yapilmasini arz | 30665 | ego araç | 3313 |
| tari hi | 27418 | rögar kapak | 3062 |
| 2018 tari | 26516 | halk otobüs | 2975 |
| hi nde | 26330 | kart sayaç | 2910 |
| lmesi ni | 24944 | sefer sayı | 2521 |
| sti yorum | 23538 | hat otobüs | 2446 |
| bi lgi | 23189 | özel halk | 2272 |
| şi kayetçi | 21797 | kanal tıkanık | 2198 |
| ni talep | 21549 | hız kesmek | 2147 |
| kayetçi yi | 21341 | yağmur su | 2092 |
| vedi li | 19120 | ego aracı | 2074 |
| sokak no | 18447 | su sayaç | 2013 |
| yi gereği | 17787 | su basmak | 1935 |

The data presented in Table 3.3 shows the statistics of the bi-gram frequency after and before preprocessing. The word groups which were meaningless before preprocessing have become sensible and consistent after preprocessing.

**Table 3.4** Tri-gram frequency comparison according to the preprocessed data

| tri-grams before preprocessing | | tri-grams after preprocessing | |
|---|---|---|---|
| ni arz ederi | 110175 | hat ego otobüs | 5830 |
| gereği ni arz | 108436 | hat ego araç | 2320 |
| yorum gereği ni | 61371 | özel halk otobüs | 2257 |
| talep edi yorum | 53285 | durak hat ego | 2170 |
| edi yorum gereği | 43684 | ego otobüs durak | 1561 |
| gereği ni yapilmasini | 30407 | hız kesmek kasis | 1434 |
| yapilmasini arz ederi | 30276 | hat ego aracı | 1334 |
| ni yapilmasini arz | 28960 | sefer sayı artırmak | 1118 |
| tari hi nde | 26291 | üst geçit asansör | 994 |
| 2018 tari hi | 23844 | su boru patlamak | 954 |
| şi kayetçi yi | 21334 | ego otobüs şoför | 846 |
| ni talep edi | 19952 | araç zarar görmek | 842 |
| vedi li kle | 17735 | ego otobüs sefer | 825 |
| yi gereği ni | 17731 | çalışmak asfalt atılmak | 790 |
| kayetçi yi gereği | 17672 | durak hat otobüs | 761 |
| sti yorum gereği | 17432 | yürümek merdiven çalışmak | 736 |
| hi nde saat | 16633 | alt yapı çalışmak | 711 |
| lmesi ni talep | 15989 | kart sayaç arıza | 700 |
| edi lmesi ni | 12157 | keçiören şehit kubilay | 680 |
| bi lgi si | 10001 | mehmet akif ersoy | 665 |

In Table 3.4, the most repetitive tri-grams are shown. The tri-grams present a more understandable statistic for us to see the related keywords. The uni-grams, bi-grams and tri-grams present the concept of the study.

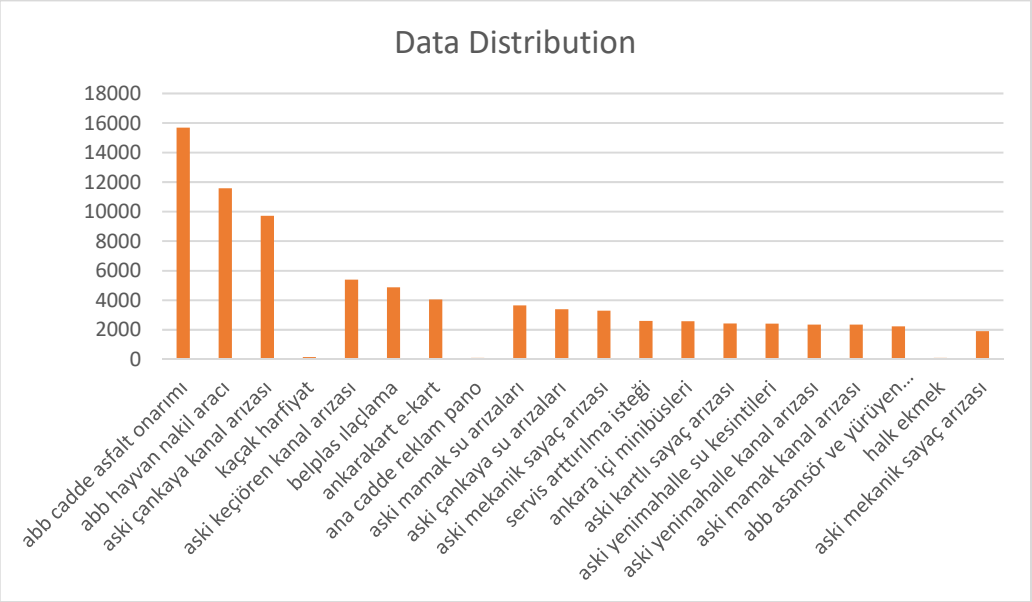


**Figure 3.11** The distribution of the data

It is seen that the classes in the corpus are excessively imbalanced. This imbalanced distribution and the fact that there is just one letter of request in some classes are predicted to affect the performance of the algorithms in a negative way. Randomly selected twenty classes in 1819 letters of request and the number of the letters are seen in Figure 3.11. To make the data more consistent, the classes containing less than 100 letters of request were taken as one class and were labeled as “others” in this study. In addition, it is also seen that some class titles refer to the same classes. For example, “YENİMAHALLE SU KESİNTİLERİ” and “ÇANKAYA SU KESİNTİLERİ” in Figure 3.11 both refer to “SU KESİNTİLERİ”. The other similar classes were also simplified. The below Figure 3.12 presents the distribution of the data after simplification. The number of the classes was reduced from 1819 to 14. Our reducing the number of the classes manually to 14 helped us detect the number of probable clusters. Thus, we gave the number of clusters required for performing clustering via the K-Means algorithm as 14. These clusters are in harmony with the classes which were simplified manually. How to use the K-Means algorithm in Sckit-learn can be seen in Appendix D.

Classification algorithms were also implemented on the new corpus and the results were compared to each other.

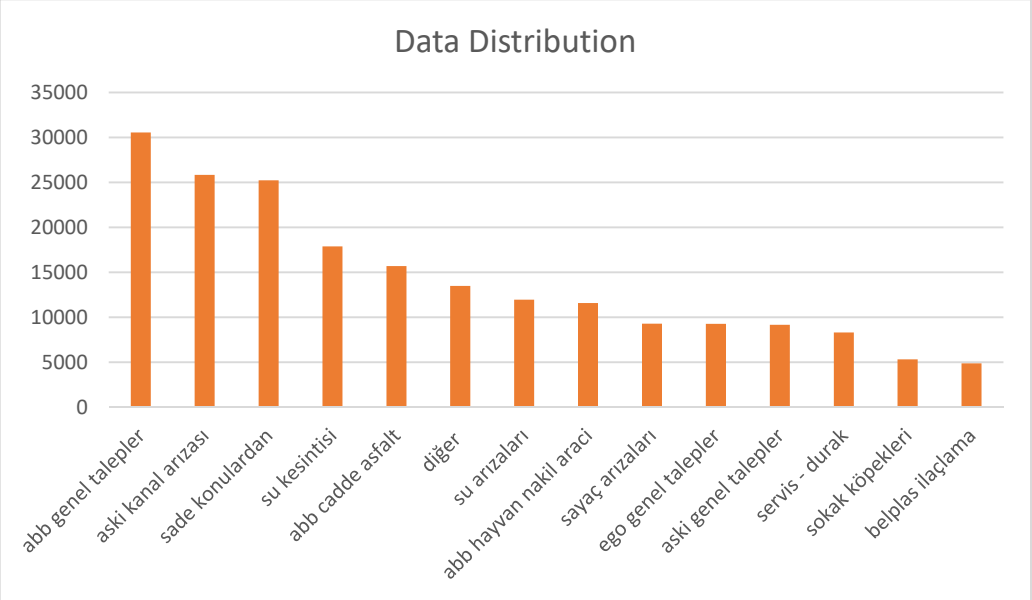


**Figure 3.12** Letter of request count after preprocessing

## 3.3 The Tools and Technologies Used

### 3.3.1 Python

Five different classification algorithms were used to generate models. To perform the training, the Python Programming Language and its libraries were used. The Python Programming Language is quite popular for scientific computing. Almost all of the studies having been done on machine learning recently are seen to have been implemented by Python.

### 3.3.2 Pandas

In an official website, the Pandas Library is defined as “an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language” [23]. The Pandas Library is used often in data science. It can be used with the aim of data analysis in three different ways:

- Convert a Python's list, dictionary or Numpy array to a Pandas data frame.
- Open a local file using Pandas, usually a CSV file, but could also be a delimited text file (like TSV), Excel, etc.
- Open a remote file or database like a CSV or a JSON on a website through a URL or read from a SQL table/database.

### 3.3.3 Matplotlib

The developers of Matplotlib define the term as in the following: "Matplotlib is a Python 2D plotting library which produces publication quality figures in a variety of hardcopy formats and interactive environments across platforms. Matplotlib can be used in Python scripts, the Python and IPython shells, the Jupyter notebook, web application servers, and four graphical user interface toolkits" [24]. The library provides us a lot of functions like drawing graphics and plotting various datasets. It is possible to visualize the data with a minimum number of code lines perfectly.

### 3.3.4 Numpy

Numpy is one of the most powerful open source Python libraries. We can think of it and Pandas as two libraries complementing each other. The two features remarked in NumPy v1.13 Manuel are shown below [25]:

- The arrays have a fixed size at the beginning while Python lists does not.
- The elements in a NumPy array are all required to be of the same data type, and thus will be the same size in memory. The exception: one can have arrays of (Python, including Numpy) objects, thereby allowing for arrays of different sized elements.

### 3.3.5 Scikit-learn

The Scikit-learn module is a Python library consisting of a lot of supervised and unsupervised machine learning algorithms. It has made the use of many well-known algorithms quite easier. It is dependent on the Numpy and the Scipy libraries [26]. All of the algorithms used in this study were implemented with the Scikit-learn Library.

### 3.3.6 Zemberek

There are a lot of available libraries especially in English for NLP. For sentences in Turkish and Turkic languages, Zemberek [27] Library is available as an open source. Zemberek has the modules of morphology, tokenization, normalization, named entity recognition, classification, language identification, language modeling and GRPC server. Via the module of Morphology, disambiguation, Turkish morphological analysis and word generation can be used; by the module of Tokenization, tokenization and sentence boundary detection can be used; with the module of Normalization, basic spell checker, word suggestion and noisy text normalization can be used; via the module of NER, Turkish named entity recognition can be used; via the module of Classification, text data can be classified. In this study, the modules of morphology and normalization are used.

### 3.3.7 Keras

Keras is a Python library which was developed for deep learning. It runs on top of Tensorflow, CNTK or Theano. It also runs on GPU or CPU. Its training time might decrease considerably if it is run on GPU. It is quite easy and fast via Keras to develop models [61]. For example, a sequential model is imported from the models in Keras via the line of code “from keras.models import Sequential”. Thus, a sequential model can be used as easily as in the line of code above. That type of model was used in our study, as well, because LSTM was to be built in layers. Keras gives us the chance to add layers to the sequential model as in the following:

```
model = Sequential()
model.add(Layer1(..., input) )
model.add(LayerN(...) )
model.compile(...)
```

### 3.3.8 Java Spring Boot

Spring Boot is a Jave-based open-source framework. It is developed by Pivotal Team. Via Spring Boot, web applications can be developed in a quite fast way. Spring Boot

makes it easy to develop stand-alone, production-grade applications that you can just run. Because it is easy to develop applications and to deploy them into production, Spring Boot was used in this study. Stand-alone applications can be developed. They include an embedded Tomcat server and do not want you to configure complicated xml [48].

### 3.3.9 React.js

React.js is a Javascript library which was developed by Facebook for building user interfaces. It is declarative and compenent-based. It is a useful choice to develop single-page web applications. It manages the objects of state thanks to the virtual document object model placed in it. Its features of having an understandable and detailed documentation and having a huge community and its popularity are our reasons for using it.

### 3.3.10 PostgreSQL

PostgreSQL is an open source object-relational database system. Its first version was developed in 1997. C programming language is used in its development. It has been developed for more than 30 years. For it to be maintained, minimum effort is required because of its stability. PostgreSQL has many advanced features;

- Table inheritance
- User-defined types
- Views, rules, subquery
- Asynchronous replication

## 3.4 Algorithms

### 3.4.1 Machine Learning

Machine learning, hence the name, means teaching a set of functions with the help of data to the machine, in other words, to the computer. People realise their learning process via inferences they gain from their experience. As for machine learning, this

experience meets us as data. On the condition that there is no data, one cannot talk of machine learning.

The computer learns by detecting regularities or patterns in the dataset. If there is no pattern, machine learning models cannot be used. There is no particular rule in machine learning. The machine parses the data to decide what to do, on its own. As we need to repeat often something new we learn, for a computer it is related to the data size to realise a better learning process.

Machine learning is defined in Faggella's survey as in the following: "Machine learning at its most basic is the practice of using algorithms to parse data, learn from it, and then make a determination or prediction about something in the world.", "Machine learning is the science of getting computers to act without being explicitly programmed" [28]. As is seen in these definitions, machine learning actually means coming to a conclusion through examining the data.

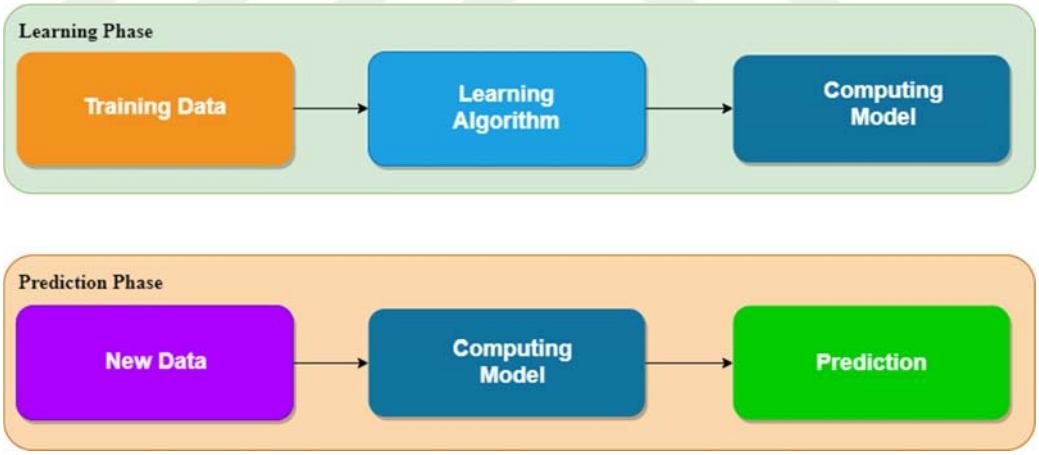


**Figure 3.13** How ML works

Data is indispensable in machine learning. As is seen in Figure 3.13, the process begins with data in machine learning processes. It has become quite easier to achieve the data thanks to the improvements in internet. As a conclusion of it, machine learning increases its popularity. The model is generated with the help of learning algorithms after acquiring the data. This model makes predictions according to the incoming data in the prediction phase and comes to a conclusion. When making predictions, this

model decides which fields are more important in the dataset according to the relationship between the former outputs and inputs.

Such examples as e-mail spam filtering, detection and diagnosis of disease, sentiment analysis and news categorization can be given for applications made via machine learning techniques.

The learning algorithms used in machine learning are divided into two categories. These learning algorithms are categorized as supervised and unsupervised according to the techniques used. In this study, supervised learning algorithms were used. In Figure 3.14, some of these algorithms are seen.

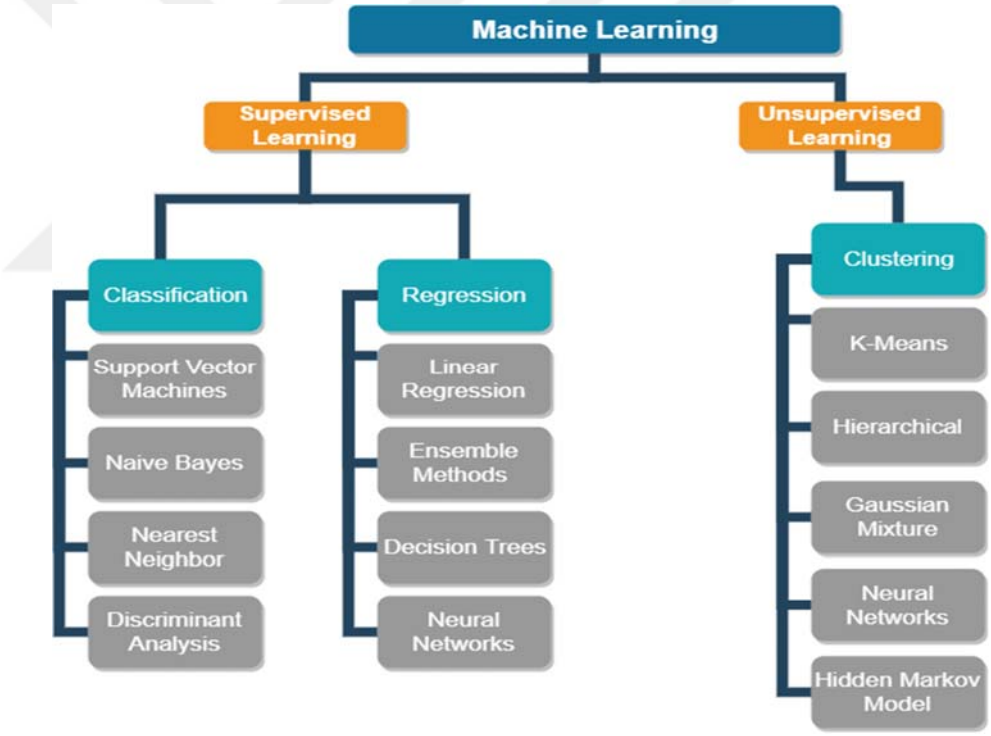


**Figure 3.14** ML methods

### *3.4.1.1 Supervised Learning*

Supervised learning is a kind of machine learning task, and “supervised” means labeled data. The ML model employes a sample dataset. Negative or positive training examples of the general category might be included in the given sample dataset. The

main goal is to acquire the definition of the category. Supervised learning algorithms find the relationships and dependencies between the input and output, and make prediction of the next output. The outputs of each input are available on the dataset trained by machine learning algorithms. In other words, the data to be used in supervised learning and its categories are attached to each other. This data is called labeled data.

Supervised learning algorithms use two different techniques to generate models: classification and regression. The technique of classification is a systematic approach to generate a classification model. Classification makes prediction of the target classes according to the former labeled data. That is why this approach is called supervised learning. Decision Tree classifier, Rule-Based classifier, Neural Network classifier, Naïve Bayesian classifier, Neuro-Fuzzy classifier and Support Vector Machines are some of these classification algorithms [29]. While the techniques of regression work on the continuous data, classification techniques work on the discrete (or categorical) data.

In this study, Naïve Bayes classifier, Support Vector Machine classifier, Random Forest classifier and Logistic Regression classifier, which are under the category of supervised learning techniques, were used with labeled text data.

#### *3.4.1.2 Naïve Bayes Classifier*

Naïve Bayes classifier is based on the Bayesian theorem. Mostly, it is used for spam filtering, sentiment analysis, recommendation systems etc. The features are independent from each other in the sample dataset to be trained with Naïve Bayes. In other words, each input might be matched with an output [49]. Naïve Bayes has been commonly used for a long time because of its easy implementability and its ability to make an accurate classification [30]. “Bayes theorem provides a way of calculating the posterior probability, P(c|x), from P(c), P(x), and P(x|c). Naive Bayes classifier assumes that the effect of the value of a predictor (x) on a given class (c) is independent of the values of other predictors. This assumption is called class conditional independence” [31].

The multinominal event model, which is one of Naïve Bayes types, is commonly used [32]. Multinominal Naïve Bayes was implemented for text classification in this study. This model is used more for document classification problems such as magazine categorization, article categorization and poem topic detection.

The use of the Bayesian approach in recent years has become more common through easy access to statistical data and the excessive increasement of this data. The Bayesian theorem was coined by Thomas Bayes in 1763 [33].

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad \textbf{(3.1)}$$

The Bayesian theorem is based on a statistical probability formula. The relevant equation is seen above, in 3.1.

$P(A|B)$: The probability of "A" being true given that B is true

$P(B|A)$: The probability of "B" being true given that A is true

$P(B)$: The probability of "B" being true

$P(A)$: The probability of "A" being true

Naïve Bayes classifier uses the Bayesian theorem, which is shown in equation 3.1. It is called naïve because the predictors and the features are thought to be independent. That is to say, a feature does not affect one another in the dataset which Naïve Bayes classifier is used in.

The Bayesian approach is one of the supervised learnings techniques. Therefore, it works with labeled data. If one wants to use Naïve Bayes classifier for text classification as a classification algorithm, the text data labeled before is required to be ready. The labeled data was sent to Naïve Bayes classifier for the topic of this study, text classification. The aim of the study is to predict which category the given text is related to using Naïve Bayes (see Figure 3.15).


**Figure 3.15** Illustration how Naïve Bayes classifier works

Naïve Bayes classifier calculates the probability of every statement for the data, and classifies the probability according to the higher one.

$$\prod_i^n P(fi|C) \quad \textbf{(3.2)}$$

In this study, if we name the features of the classified text data as n; the above equation 3.2 can be used to find which class the data belongs to using the Bayesian theorem when new text with an unknown class comes.

#### *3.4.1.3 Support Vector Machine*

Support Vector Machine (SVM), which is one of supervised learning algorithms, is a kernel-based machine learning algorithm used to solve both classification and regression problems [50]. It is used in such areas as handwriting digit recognition, object recognition, speaker identification and face detections in images. It began to be used to solve classification problems in 1990s [34]. SVM, to put it simply, determines the borders between the data classes using support vectors.

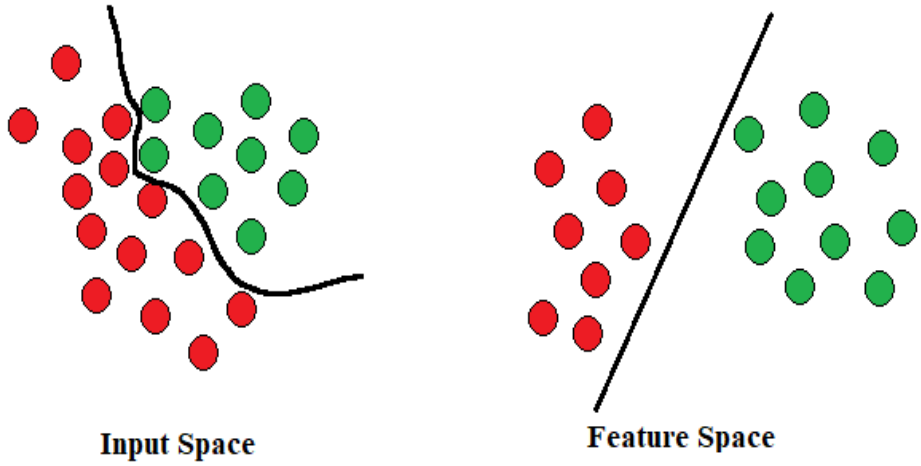


**Figure 3.16** Illustration of SVM

The illustration above consists of input and feature spaces. The data is transformed into the dot notation form using mathematical functions known as kernel. The kernel mapping functions are quite powerful because they support SVM even in so complicated circumstances to realize the separation process. The feature space, which

is divided into two parts by a linear, shows the division of the green and red objects linearly.

SVM works to categorize the data in an optimal way by an n-dimensional hyperplane. It uses the closest one to the data to choose the most suitable one among the hyperplanes. The distance between the data and the hyperplane is called margin. As for finding the most ideal hyperplane, it uses support vectors. The margin is expanded by using these support vectors, and a more consistent hyperplane is chosen. (see Figure 3.17) The dimension of the hyperplane depends on the dimension of the input features. There is a two-dimensional hyperplane in Figure 3.16 and Figure 3.17. The hyperplane refers to just a line. If the number of the features increases, the number of the dimension also increases.

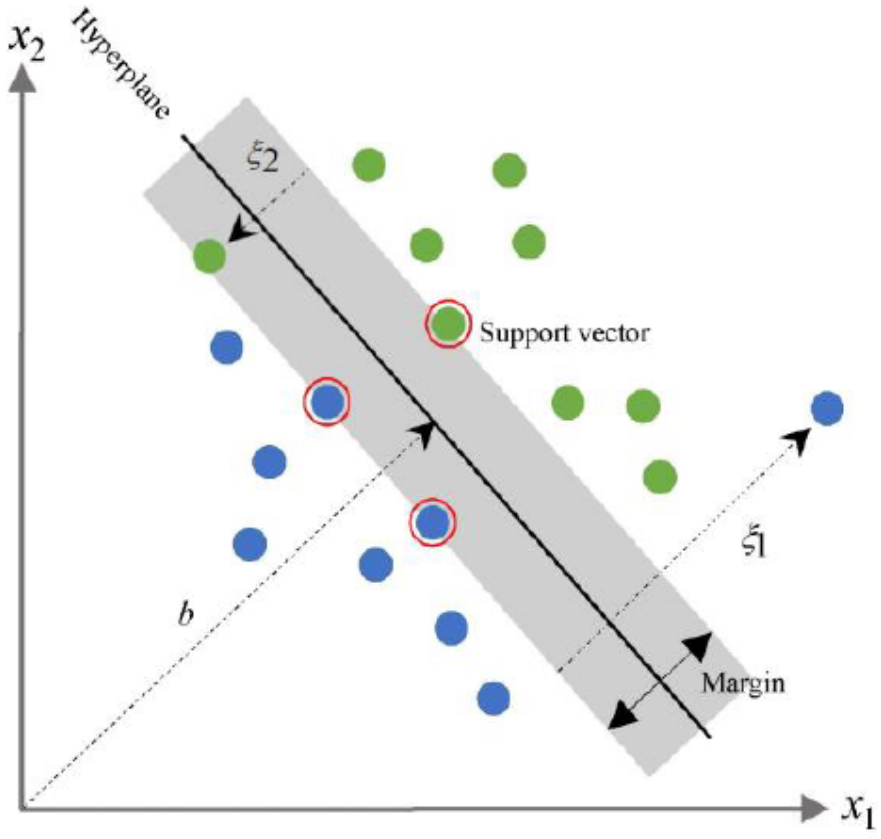


**Figure 3.17** Margin, hyperplane and support vector [35]

#### *3.4.1.4 Logistic Regression*

Logistic Regression is one of the algorithms commonly used and counted as basic among machine learning algorithms. It is called so because it performs the classification process drawing a logarithmical function. It is a regression algorithm because it makes prediction on the continuous outcome [53]. Logistic Regression is ideal for datasets which consist of just two categories. While its use like [10] is seen in just biological sciences in 1900s, it is used in many branches today.

There are three main methods called binary, multinomial and ordinal logistic regression. Binary Logistic Regression is used on data dependent to each other. This data has binary results like yes/no and does/do. Ordinal Logistic Regression is applied when the result variables are ordinal and the number of the classes must be at least three. For these classes, light/medium/strong or the numbers between one and three could be given as examples. When there are three or more classes, Multinomial Logistic Regression is used. These classes do not need to be ordinal [36]. In this study, as well, Multinomial Logistic Regression was used because the number of the classes is more than three and the dataset is not ordinal. The function called sigmoid is used to predict which class particular data belongs to when studying on binary data (see Figure 3.18).

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad \textbf{(3.3)}$$

Where $\sigma$(z) is an output between 0 and 1, (z) is the input of the function in Equation 3.3. As for e, it is the base of the natural logarithm.

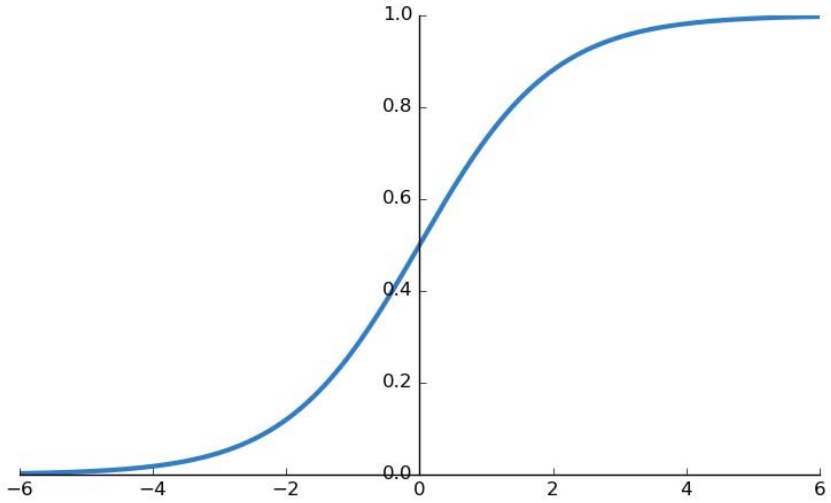


**Figure 3.18** The sigmoid function [37]

As is seen in Figure 3.18, the value of the sigmoid function is between zero and one. Figure 3.18 shows that it is quite suitable for binary mapping. When 'z' goes to infinity, the prediction of 'y' will be 1. When 'z' goes to negative infinity, the prediction of 'y' will be 0.

$$\sigma(z) = \frac{e^z}{\sum_{J=0}^{k} e^z} \quad \textbf{(3.4)}$$

The softmax function is used when processing on a dataset consisting of three or more classes. By the above equation 3.4, the softmax function is described.

The softmax function examines the probability of each class on the probable other classes. When performing multiclass classification, such techniques as one-versus-all and one-versus-one are used [14]. In the softmax function, the total value of the probabilities must be 1.

#### *3.4.1.5 Random Forest Classifier*

Random Forest was coined by Leo Breiman, who was affected by Amit and Geman's studies. Random Forest can be used for classification or regression [13]. Random Forest classifier, an ensemble approach, consists of decision trees. The ensemble

approach refers to using one or more than one same or different algorithms to solve a problem. Random Forest classifier consists of decision trees. The fact that decision trees can be easily implemented and are flexible and interpretable causes us to meet datamining very often. The implementation of Random Forest classifier is easy, as well, and its prediction and training take a short time. It is used in such areas as image classification, detecting fraudulent cases in banking systems, recommendation engines, feature selection and multiclass classification.

A decision is made according to the majority voting after all the decision trees' running is completed, and the classification is performed (see Figure 3.19). This type of algorithm is called 'random' because it selects a random subset from the training set.

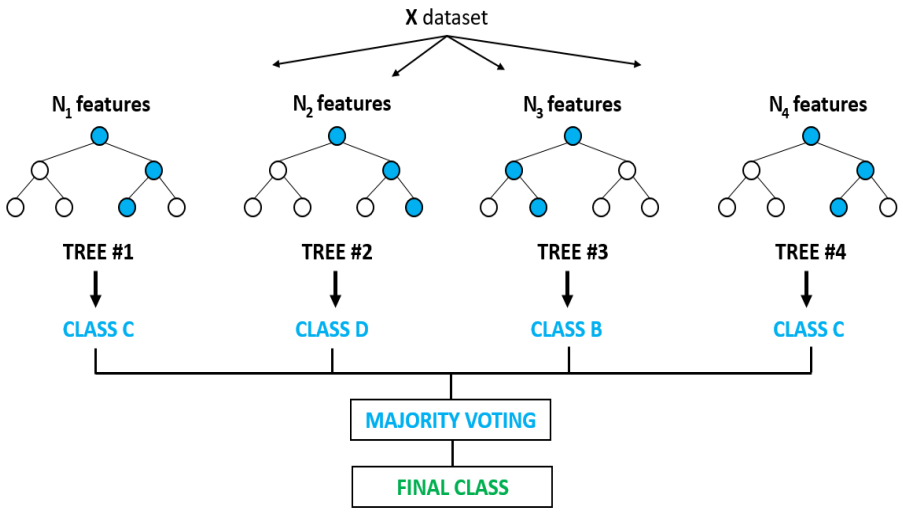


**Figure 3.19** Illustration of Random Forest prediction [38]

To put Figure 3.19 simply, the features 'N1','N2','N3','N4' are data selected randomly from the dataset X, and any data could belong to B, C, D. The class having the highest probability among the classes predicted by processing these N1, N2, N3, N4 features, which are selected randomly, via the decision trees, gives us the final class.

### *3.4.1.6 Unsupervised Learning*

Computers themselves perform the learning process by finding regularities in the input. In the dataset, there is neither a specific output nor data related. There is no labeled data. The machine generates models finding a resemblance and a pattern among the inputs. “The two unsupervised learning tasks we will explore are clustering the data into groups by similarity and reducing dimensionality to compress the data while maintaining its structure and usefulness” [39].

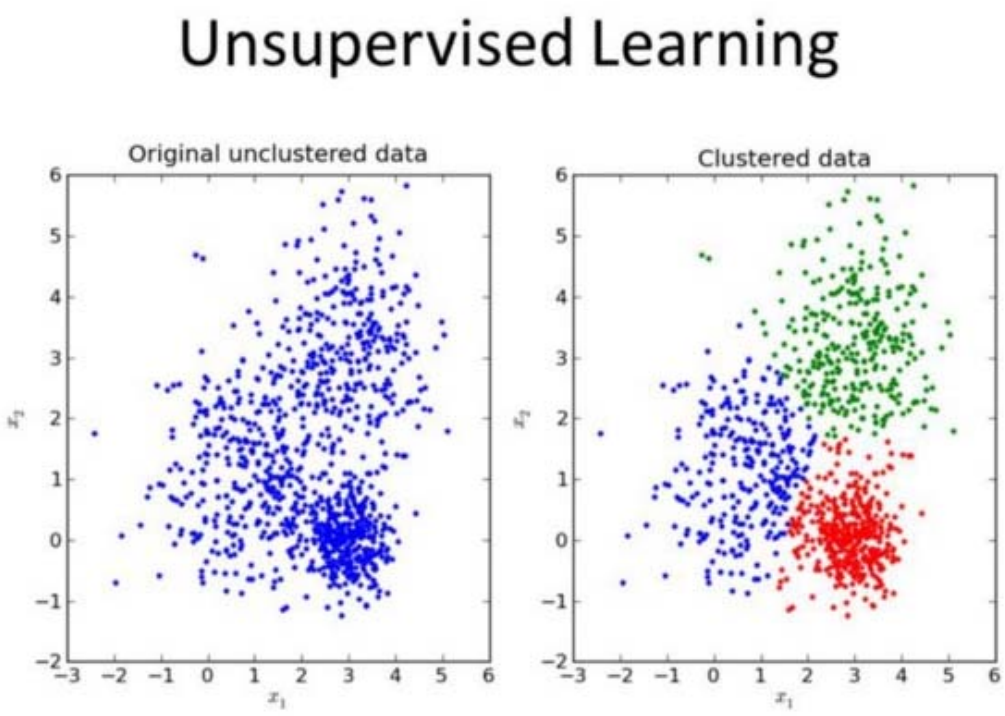


**Figure 3.20** Illustration of clustering [40]

Unsupervised learning looks for a relation between the datasets. It might be a negative or positive relation. In other words, if there is a resemblance or difference between the datasets, unsupervised learning detects this pattern. As is seen in Figure 3.20, clustered data was gained performing clustering on the similar data in the original datasets. It is called unsupervised because there is no sample dataset or teacher. Some of widely used unsupervised learning algorithms are K-Means Clustering, Hidden Markov

Model, Singular-Value Decomposition (SVD) and Principal Component Analysis (PCA).

#### *3.4.1.7 The K-Means Algorithm*

K-Means, which is an unsupervised learning algorithm, performs clustering on datasets. It is a data analysis technique often used for gaining information about the structure of the data. A cluster is a group of similar entities. The clustering techniques are widely applied in many applications such as artificial intelligence, biology, customer relationship management, data mining, information retrieval, image processing etc. In K-Means, each cluster represents the average value of the objects in the cluster. The K-Means algorithm takes the corpus and the number of desired clusters as the input parameter. The cluster set returns as the output [51].

#### *3.4.1.8 Long Short-Term Memory*

In feedforward networks, information is processed only forward. The given inputs reach the output layer through the other layers. The output values obtained as a result of the training are compared with the expected output values, and error values are obtained. Recurrent networks, which are deep learning algorithms, use their output as their input in the next step. RNN uses their internal memory to process sequences of input. Recurrent neural networks have a backward connection between hidden layers. Time series analysis, NLP, text classification and speech recognition can be given as examples of RNN applications. In neural networks, the output coming out of the hidden layer goes back to the network.

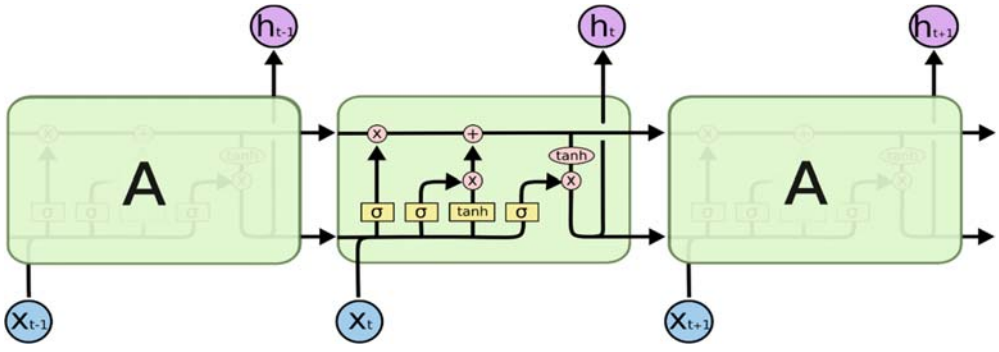


**Figure 3.21** LSTM layers [63]

As for LSTM, it is a special kind of RNN which has an efficiently extended memory. It was developed by Sepp Hochreiter and Juergen Schmidhuber in 1997 [56]. There are more layers in LSTM, not just one neural network layer, and these layers are called gates (see Figure 3.21). LSTM decides what to write in the memory thanks to these gates. Thanks to the memory, the previous information can be obtained and conveyed to the next one. Which data among this information to use is determined by training. RNN with LSTM is an effective and scalable model for several learning problems.

### 3.4.2 Text Classification

Text classification aims to detect which category the documents in a corpus are related to. To define it formally, "if $d_i$ is a document of the entire set of documents D and $\{c_1, c_2, ,\ldots, c_n\}$ is the set of all the categories, then text classification assigns one category $c_j$ to a document $d_i$" [41]. The size of the text data increasing in the digital platforms makes this area more and more popular. Via text classification, such applications as spam mail detection, sentiment analysis, the detection of which topic a paper is related to, movie class detection and customer comment analysis in e-commerce systems are developed in many fields. The concerned text might be labeled to just one category or more than one category. For instance, a magazine article might be labeled to both sports and health classes. If the text data is related to one category, this data is called "single-label"; if it is related to more than one category, this data is called "multi-label" [42]. In this study, as well, the results were evaluated by using different algorithms to detect which department the letters of request having come to an organization are related to.

As is seen in [43], 80% of all the data in the world is unstructured and text data is quite a common type. Both organizations and firms can take a more consistent process or improve their present process by transforming this data into sensible information.

#### *3.4.2.1 Documents*

As in every machine learning tasks, we need a sample dataset for the data to be classified in text classification, as well. Sentimental analysis on tweets sent via Twitter in [44], news categorization of the news in [45] could be given as examples for text

classification studies. The tweets and news used in both these studies could be given as indispensable documents in text classification processes. In addition, email, legal documents, web pages, chat conversations and social media messages are good documents to apply text classification process on.

### *3.4.2.2 Preprocessing*

The data was taken under the title of documents. Preprocessing is required for training the documents in a consistent way via machine learning algorithms. There might be misspellings and unnecessary characters or words in the data if it directly comes from users (e.g. social media messages, email or chat conversations). These misspellings might cause the text classification algorithms to work in a wrong way. Because the algorithm will take the different spellings for the same word as different words, there might be different results [46].

Natural Language Processing (NLP), which is a subfield of Artificial Intelligence (AI), examines the interaction between humans and computers. NLP techniques can be used in need in preprocessing.

We start with the cleaning step for improving the data. Firstly, all the documents are transformed into lowercases. Then all the numbers, special characters and punctuation marks placed in the documents are omitted. Lastly, the stop words are removed (see Figure 3.22). All these steps are required for the data to work in a more consistent way and to give accurate results. Thanks to preprocessing, the documents are deduplicated.


**Figure 3.22** Cleaning process

Documents come from users. That is why, words contained in documents might be wrong or inadequate. Via normalization, these words are corrected. Then, the stemming step is taken for the data to be deduplicated in all the documents. These steps are called Morphological Analysis in NLP.

### *3.4.2.3 Feature Extraction*

We cannot send the preprocessed data directly to machine learning algorithms. Most machine learning algorithms need feature vectors to work. That is why, before using machine learning algorithms, the dataset requires to be transformed into feature vectors for classification. The model “Bag-of-Words” is one of the most common techniques. This model is applied with the calculation of the number of the terms in a document. Such features as term position and ordering of words are overlooked. In the bag-of-words model, the methods of occurrence of word as feature and the number of word occurrences as feature value are given as examples (see Figure 3.23 and Figure 3.24). In this study, TF-IDF, which is based on the bag-of-word model, was used.

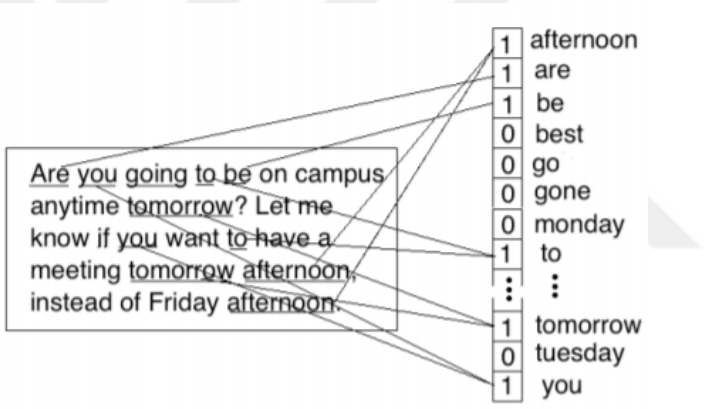


**Figure 3.23** Word occurance as feature [47]

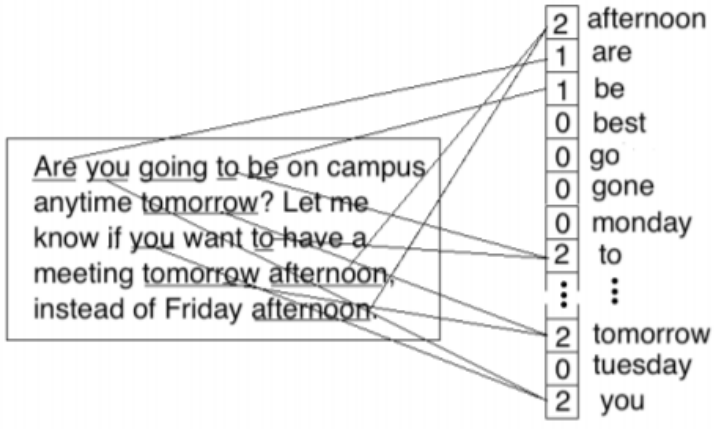


**Figure 3.24** The number of word occurances as feature [47]

#### *3.4.2.4 Term Frequency - Inverse Document Frequency (TF - IDF)*

Term Frequency, which is the long form of TF, deals with how many times a word is used in a document. The important thing for TF is the frequency of a word. Inverse Document Frequency (IDF) is used to detect which word is more important. When calculating TF, the importance of all the words examined is counted as equal. Whether the words examined are stop words or insignificant conjunctions is overlooked. This problem is solved by combining TF and IDF [54]. Via TF-IDF, the importance of the words in the documents contained in the corpus is determined. The equation below shows the formula of TF-IDF.

$$tfidf_{i,j} = tf_{i,j}\ x\ \log\left(\frac{N}{df_i}\right) \quad \textbf{(3.5)}$$

$tf_{i,j}$ = total number of occurences of i in j
$df_i$ = total number of documents containing i
N = total number of documents

# CHAPTER 4

## EXPERIMENTAL RESULTS

### 4.1 Performance Metrics

In the comparing of machine learning algorithms according to their prediction performance, the values of accuracy, recall, precision and F1-Score are commonly used. We evaluated the results of the model which was trained by LSTM considering the values of accuracy, loss, validation accuracy and validation loss acquired with the data of train and validation. Then, we chose the values to show whether the model is overfitting or not. If the training loss is lower than the validation loss, the model is overfitting. If the training loss is higher than the validation loss, the model is underfitting. If they are equal or very close to each other, it can be said that an accurately fitting model has been developed. Accuracy meets us as a simple and common evaluation metric. Accuracy helps measuring the classification performance of the algorithm. It gives us the probability of prediction when new data comes. It is also useful to look at two more metrics called recall and precision when evaluating the performance of the machine learning algorithm along with its accuracy.

True positive: Samples correctly classified as positive
False negative: Samples incorrectly classified as negative
False positive: Samples incorrectly classified as positive
True negative: Samples correctly classified as negative
The formulas for accuracy, precision, recall and F1-score are given in the equations below.

$$Accuracy = \frac{T_{positive} + T_{negative}}{T_{positive} + T_{negative} + F_{positive} + F_{negative}} \quad \textbf{(4.1)}$$

$$Precision = \frac{T_{positive}}{T_{positive} + F_{positive}} \quad \textbf{(4.2)}$$

$$Recall = \frac{T_{positive}}{T_{positive} + F_{negative}} \quad \textbf{(4.3)}$$

$$F1 - Score = 2 \text{ x } \frac{Precision \; x \; Recall}{\text{Precision+Recall}} \quad \textbf{(4.4)}$$

The value of recall tells us the rate of detecting the class correctly. Precision, on the other hand, gives us how many of the classes predicted are accurately true. F1-score is calculated according to both of them.

Accuracy itself is not enough especially in datasets of which some classes have lots of data. For example, the dataset in cancer detection, which is quite a popular working area especially in machine learning studies, consists of two classes. Healthy people constitute a large part of the dataset. The algorithm will usually predict that upcoming data belongs to healthy people. Taking our dataset into consideration, it is not enough to analyze only the accuracy because of its being imbalanced.

## 4.2 Experimental Results

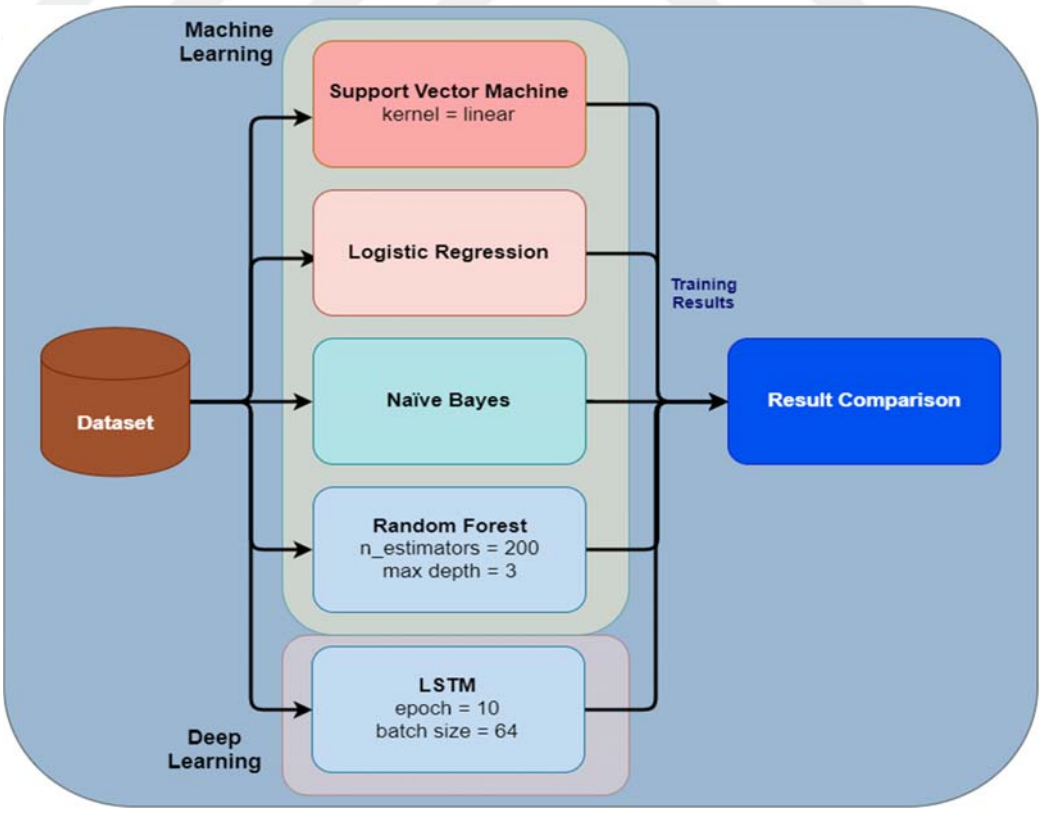


**Figure 4.1** Model training

The dataset was trained via machine learning algorithms to generate models after and before preprocessing. 25% of the dataset, consisting of 225239 letters of request, was used as test data and the rest as training data. In Figure 4.1, the algorithms used are given. Random Forest algorithm is an ensemble algorithm. We can think of it as a forest containing lots of decision trees. The number of the decision trees to be contained when using Random Forest classifier with Scikit-learn needs to be given as the parameter. In this study, a forest consisting of 200 decision trees was used. In addition, the maximum depth was given as three. Via SVM, Naïve Bayes and Logistic Regression, which are contained in Sckit-learn, models were generated. In SVM, the linear kernel was used. The models acquired were tested with the test data and its values of precision, recall, F1-score and accuracy were compared to each other.

**Table 4.1** Results before preprocessing

| Algorithm | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| SVM | 59% | 60% | 64% | 64% |
| LR | 56% | 50% | 54% | 52% |
| NB | 35% | 29% | 29% | 29% |
| RF | 10% | 9% | 12% | 10% |

Table 4.1 shows the results acquired with the raw data. SVM is seen to have the highest accuracy. Along with it, its values of precision and recall are also higher than the other algorithms'. High recall value returns most of the relevant results. The percentages are given above, in Table 4.1. The accuracy value of Random Forest classifier is seen to be quite low when the algorithms are worked with the raw data. In addition, Random Forest is seen not to work well with the data used in the study, as well. Along with the accuracy value, the value of F1-score is also quite low. SVM and Logistic Regression have a more than 50% accuracy despite the fact that the data is raw. Taking that the data used in this study is imbalanced into consideration, it is needed to look at the values of recall and F1-score along with accuracy to be able to evaluate the performance of the machine learning algorithms.

**Table 4.2** Results after preprocessing without the simplification step

| Algorithm | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| SVM | 61% | 66% | 73% | 68% |
| LR | 57% | 60% | 70% | 61% |
| NB | 36% | 47% | 46% | 33% |
| RF | 10% | 20% | 14% | 7% |

Table 4.2 shows the results acquired via the algorithms and the preprocessed data without the simplification step. The high results in the values of accuracy, precision and recall of SVM are the results attention grabbing at first. It can be said that Random Forest showed the worst performance.

**Table 4.3** Results after preprocessing with the simplification step

| Algorithm | Accuracy | Precision | Recall | F1-Score |
|---|---|---|---|---|
| SVM | 78% | 77% | 78% | 77% |
| LR | 76% | 76% | 77% | 75% |
| NB | 70% | 72% | 70% | 66% |
| RF | 18% | 33% | 18% | 7% |

In Table 4.3, the results acquired after preprocessing with the simplified and balanced data in the corpus are shown. These values show that SVM is the fittest machine learning algorithm for our dataset. In Table 4.1, Table 4.2 and Table 4.3, the values of accuracy, precision and F1-score are given in a detailed way.

After the imbalanced data was stabilized, the machine learning algorithms were worked. In Table 4.3, the results of the algorithm worked with the dataset containing combined classes according to the threshold value are shown. According to the results, the accuracy of SVM is 78%, the precision is 77% and the recall is 78% with a great performance. Comparing the imbalanced data in Table 4.2 to the results, a 8% increase in the accuracy of Random Forest, a 13% increase in its precision and 4% in the recall value are seen. In addition to Random Forest, a significant increase in the performance of the other algorithms is also visible. Decision trees work better in binary classification. Because the data used in this study belongs to more than two classes, the performance of Random Forest is quite low.

In Table 4.3, the results of the algorithms worked on the dataset acquired with the simplification of the classes are shown. Comparing Table 4.2 to Table 4.3, it is seen

that the balanced data distribution and the consistence of the number of the classes affect the performance of the algorithms positively.

As for the results acquired from the model trained with LSTM, it can be said that we acquired a better result with regard to accuracy than we did with traditional machine learning algorithms. We gave parameters to LSTM when training our model using Keras. We added the embedding layer used in text data to set the input layer of the model. The embedding layer provides a dense representation of words. The embedding function takes the number of inputs, output dimension and input dimension as parameters. In our study, we used the softmax function as the activation function because our dataset consists of multiclasses. The output layer was dependent on the number of classes. Both after and before the simplification step, the given parameter changed. Because we had a dataset consisting of multiclasses, we set the loss function as categorical crossentropy. The epoch size was given as 10, the batch size was given as 64. Also, we added the spatial dropout 1D layer and gave (probability of setting outputs from the hidden layer to zero) 0.2 rate to it. It drops entire 1D feature maps instead of individual elements. Srivastava et al. in [62] mention the use of dropout for preventing overfitting. In our study, we added the spatial dropout to the model to reduce overfitting. Adam (Adaptive Moment Estimation) was used as the optimizer because of its features of low memory requirement, easy implementation and working well with large datasets. When using the fit function of the model, the input data, the labelled data belonging to the input data, the validation split and callback parameters were given. We stopped the training via early stopping when the training went worse. Our LSTM layer had 100 memory units. The source codes are available in Appendix C. Lastly, we tested the letters of request when both preprocessed and raw with the given parameters.

**Table 4.4** Results before preprocessing via LSTM

| Epoch | Loss | Accuracy | Val. Loss | Val. Accuracy |
|---|---|---|---|---|
| Epoch 1 | 4.3711 | 0.1892 | 3.2535 | 0.3197 |
| Epoch 2 | 2.8518 | 0.3879 | 2.3750 | 0.4741 |
| Epoch 3 | 2.1148 | 0.5220 | 1.9185 | 0.5641 |
| Epoch 4 | 1.6860 | 0.6024 | 1.6889 | 0.6100 |
| Epoch 5 | 1.4133 | 0.6516 | 1.5826 | 0.6310 |
| Epoch 6 | 1.2277 | 0.6864 | 1.5441 | 0.6355 |
| Epoch 7 | 1.0811 | 0.7158 | 1.5436 | 0.6416 |
| Epoch 8 | 0.9675 | 0.7402 | 1.5473 | 0.6410 |
| Epoch 9 | 0.8663 | 0.7627 | 1.5621 | 0.6415 |
| Epoch 10 | 0.7816 | 0.7832 | 1.5907 | 0.6429 |

**Table 4.5** Results after preprocessing without simplification via LSTM

| Epoch | Loss | Accuracy | Val. Loss | Val. Accuracy |
|---|---|---|---|---|
| Epoch 1 | 3.8299 | 0.2713 | 2.6550 | 0.4469 |
| Epoch 2 | 2.3179 | 0.4955 | 2.0117 | 0.5415 |
| Epoch 3 | 1.8774 | 0.5608 | 1.7883 | 0.5775 |
| Epoch 4 | 1.6621 | 0.5945 | 1.6821 | 0.5933 |
| Epoch 5 | 1.5329 | 0.6138 | 1.6271 | 0.6057 |
| Epoch 6 | 1.4391 | 0.6306 | 1.5955 | 0.6082 |
| Epoch 7 | 1.3630 | 0.6432 | 1.5782 | 0.6134 |
| Epoch 8 | 1.3030 | 0.6552 | 1.5734 | 0.6143 |
| Epoch 9 | 1.2497 | 0.6662 | 1.5723 | 0.6146 |
| Epoch 10 | 1.2071 | 0.6754 | 1.5751 | 0.6169 |

**Table 4.6** Results after preprocessing via LSTM

| Epoch | Loss | Accuracy | Val. Loss | Val. Accuracy |
|---|---|---|---|---|
| Epoch 1 | 0.8921 | 0.6979 | 0.7169 | 0.7474 |
| Epoch 2 | 0.6611 | 0.7684 | 0.6633 | 0.7626 |
| Epoch 3 | 0.5996 | 0.7880 | 0.6521 | 0.7702 |
| Epoch 4 | 0.5570 | 0.8021 | 0.6503 | 0.7716 |
| Epoch 5 | 0.5231 | 0.8136 | 0.6511 | 0.7693 |
| Epoch 6 | 0.4949 | 0.8234 | 0.6594 | 0.7705 |
| Epoch 7 | 0.4694 | 0.8426 | 0.6729 | 0.7719 |

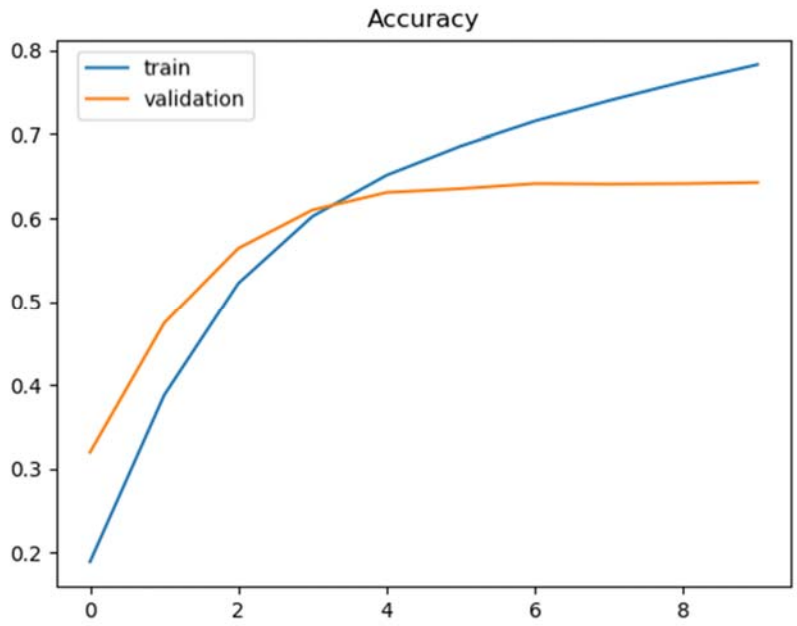


**Figure 4.2** Accuracy plot before preprocessing

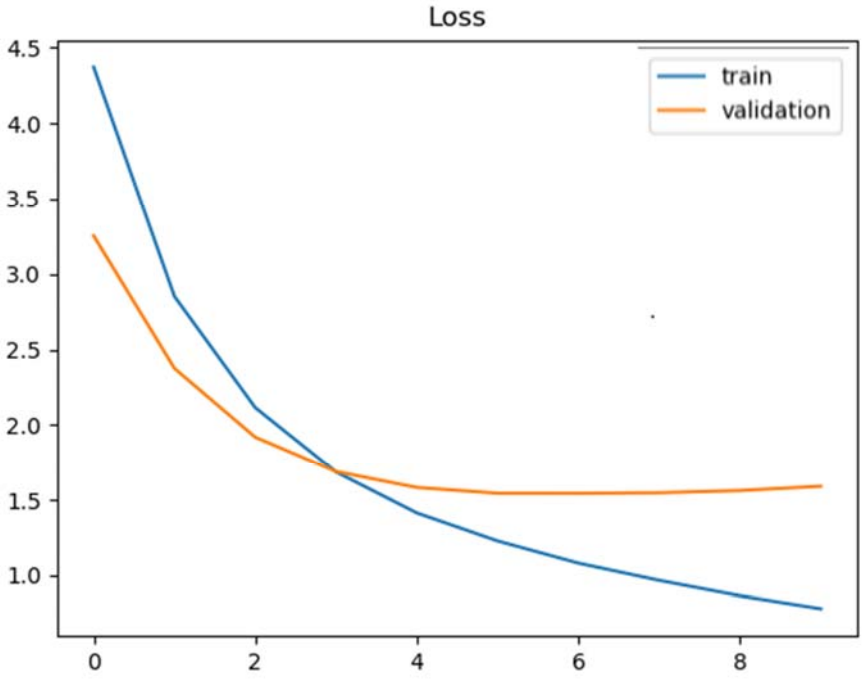


**Figure 4.3** Loss plot before preprocessing

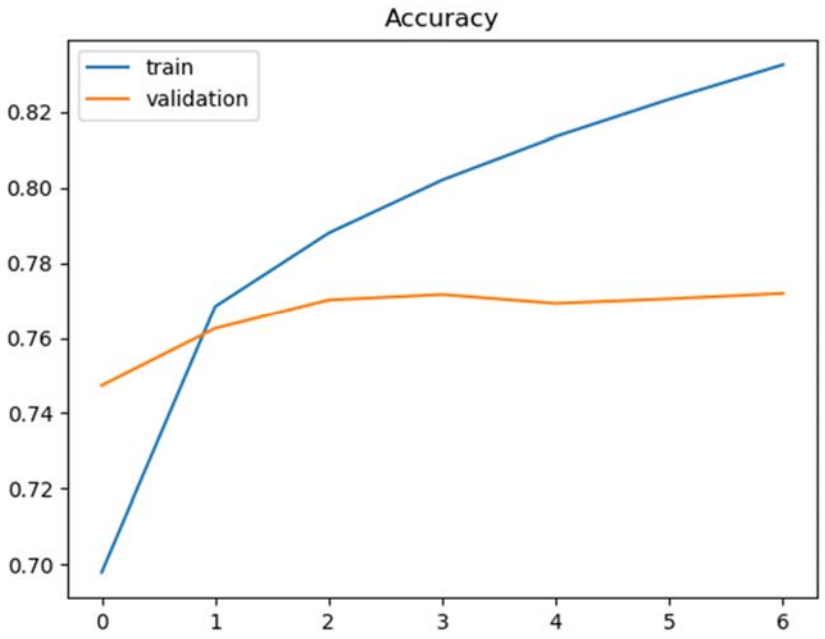


**Figure 4.4** Accuracy plot after preprocessing

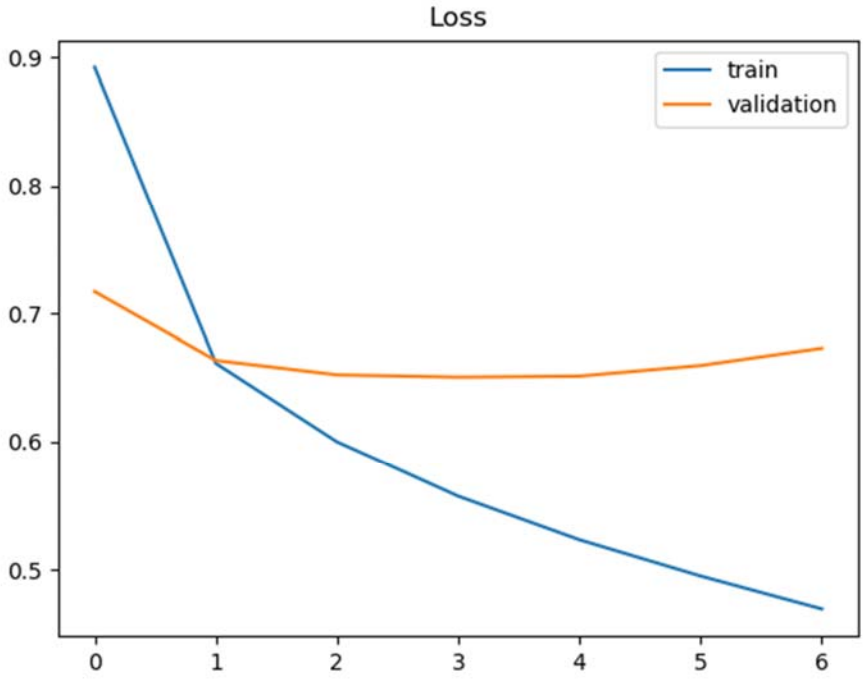


**Figure 4.5** Loss plot after preprocessing

It is seen in Table 4.4 that the accuracy value of LSTM which was worked with the raw data is 78% in the last epoch. The decrease in the values of loss and validation loss and the increase in the values of validation accuracy and accuracy coming with the increase in the number of epochs show us there is not such a big overfitting problem in the training of the model. In Figure 4.2 and Figure 4.4, the accuracy and loss graphics of the results having been acquired from the model which was trained before preprocessing are given. The model is seen to have a little overfitting problem.

In Table 4.6, the results having been acquired from the model which was trained with the preprocessed data are given. Ten epochs was given to Keras as the parameter, but the training was completed after the seventh epoch. Early stopping, given as the parameter, detected that there was no improvement in the model and stopped the training. Therefore, the training was completed after the sevent epoch instead of the tenth epoch. The accuracy value of the model trained with the preprocessed data became 84%. In Figure 4.4 and Figure 4.5, there are seen to be deviations in the graphics of train loss, validation loss, train accuracy and validation accuracy as the number of epochs increases. The model which was trained can be said to have a little overfitting problem.

The performance of the algorithms used in this study was evaluated not only with regard to accuracy, recall and precision, but also with regard to their duration of training. The system used for the experiment has 32 GB RAM, Intel® Core™ i7-6820HQ 2.70GHz CPU. The algorithms generated the models by using this system. The speed of each algorithm was different to each other with regard to the given dataset. In Table 4.7 below, the training time of each algorithm for three different statements of the dataset is given as seconds. According to the table, the training time of all the algorithms worked with the raw dataset is seen to be long. The fastest training is seen to be with the preprocessed dataset. As is seen in the table, Naïve Bayes is the fastest algorithm with regard to training time. As Korde and Mahender state in [7], if data is not clean, not only the values of the algorithm like precision and accuracy but also its training time is affected by that.

**Table 4.7** Training time comparison

| Dataset | SVM | RF | NB | LR | LSTM |
|---|---|---|---|---|---|
| Raw | 46943.523 | 36.727 | 41.663 | 5850.307 | 60214.325 |
| Preprocessed without simplification step | 9879.506 | 23.387 | 20.232 | 5418.467 | 58879.152 |
| Preprocessed | 8865.088 | 7.445 | 0.227 | 39.614 | 10426.658 |

The calculations, the iteration numbers and the number of classes, which are all required for the algorithms to complete their learning process, are in direct proportion. That is why, it is seen that the learning time becomes longer as the number of classes and the imbalance of the dataset increase. Just as in humans, the learning time of machines becomes longer as the number of classes increases. The learning time of algorithms depend on the complexity degree of the calculations.

# CHAPTER 5

# CONCLUSION AND FUTURE WORKS

## 5.1 Conclusion

In this study, five different classification algorithms were worked with the Turkish letters of request having come to an organization, and the results were examined in a detailed way. The algorithms were first worked when the data was raw, secondly when the data was preprocessed with the simplification step and lastly without the simplification step. Along with the values of accuracy, precision, recall and F1-score, training time was also examined during the comparison.

To conclude, the results given in Table 4.1, Table 4.2, Table 4.3 and Table 4.5 show that preprocessing without the simplification step did not have a notable positive impact on the study. It can be said that the order of the steps taken during the preprocessing is among what affect the performance of the algorithms. The reducing of the 1819 classes in the dataset to 14 classes helped the algorithm work better. It is seen in this study that balanced data in text classification and the harmony between the number of classes and the number of training sets are quite important factors affecting the performance of the algorithm.

In the comparison of Table 4.2 and Table 4.3, it is seen that the simplification step is the step having the greatest impact on the results among the preprocessing steps. It is concluded that stop word cleaning affected the result neither in a positive way nor in a negative way. On the other hand, it can be concluded that the TF-IDF method used in the step of feature extraction has the greatest effect on the result. In the stop word cleaning step, the most repetitive words (see Figure 3.3) in the corpus were removed. As a conclusion, the algorithms were worked with the clean data and the results were acquired. As for TF-IDF, it was used to examine the words in the corpus. To put TF-IDF simply, it determines a numerical value for each word and gives a low value to the most repetitive words. In fact, it can be said that the words we removed because they repeated a lot in the stop word cleaning step already would have had a low value

if they had been processed with TF-IDF without being removed. Therefore, there is not a big difference between Table 4.1 and Table 4.2. When the results are compared to the studies presented in the literature review part, it can be said that the results in our study are quite successful, with 84% performance. Despite the model trained with traditional machine learning algorithms, the model trained with LSTM can be said to work better both with regard to training time and accuracy. It is aimed to extend the dataset for the model to give better results with regard to accuracy via the SaaS application developed in this study.

What is more, the text classification method used in this study can be used for not only Turkish data but also the other languages. When doing a similar study, one needs to use a technique specific to their own language in the step of morphological analysis. For instance, for an Arabic study, the morphological analysis techniques mentioned by Al-Harbi and his friend in [55] can be used.

## 5.2 Future Works

In NLP tasks, it is very important for the data to be processed. The fact that data comes from natural languages (from humans) might cause an excessive dirt on the data. In addition, it should not be overlooked that data used in supervised learning algorithms needs to be labeled consistently for the trained model to give consistent results. More efficient results can be acquired from the model to be trained with data labeled in a more consistent way. The model to be built with data to be obtained via the web application we developed might improve the results. In other words, when the dataset is extended with new data to be obtained via the SaaS application developed in this study, an LSTM model to give better results with regard to accuracy might be developed. It is also possible to reduce the time to be spent for preprocessing with clean data.

# REFERENCES


[1] Rouse, M. (2019, 02). techtarget. Retrieved 09 27, 2019, from https://searchcustomerexperience.techtarget.com/definition/CRM-customer-relationship-management.

[2] Duque, J., Filipe, V. M., Varajão, J., & Dominguez, C. (2013). Implementation of CRM systems in Portuguese Municipalities. Local Government Studies.

[3] Pranckevicius, T., & Marcinkevicius, V. (2017). Comparison of Naïve Bayes, Random Forest, Decision Tree, Support Vector Machines, and Logistic Regression Classifiers for Text Reviews Classification. Baltic J. Modern Computing, 5(2), 221-232.

[4] Gamallo, P., & Garcia, M. (2014). Citius: A Naive-Bayes Strategy for Sentiment Analysis on English Tweets. SemEval@COLING. Dublin.

[5] Purohit, A., Atre, D., Jaswani, P., & Asawara, P. (2015). Text Classification in Data Mining. *International Journal of Scientific and Research Publications*, 5(6), 381-385.

[6] Bhumika, Sehra, S. S., & Nayyar, A. (2013). A Review Paper on Algorithms Used for Text Classification. I*nternational Journal of Application or Innovation in Engineering & Management (IJAIEM)*, 2(3), 90-99.

[7] Korde, V., & Mahender, C. (2012). Text Classification and Classifiers: A Survey. *International Journal of Artificial Intelligence & Applications (IJAIA)*, 85-99.

[8] El-Halees, A. M. (2007). Arabic Text Classification Using Maximum Entropy. *The Islamic University Journal (Series of Natural Studies and Engineering)*, 15(1), 157-167.

[9] DAI, Y., Guo, W., Chen, X., & Zhang, Z. (2018). Relation Classification via LSTMs Based on Sequence and Tree Structure. *IEEE Access*, 6, 64927-64937.

[10] Cessie, S. L., & Houwelingen, J. V. (1992). *Ridge Estimators in Logistic Regression. Wiley for the Royal Statistical Society*, 41(1), 191-201.

[11] Mohammad, A. H., Alwada'n, T., & Al-Momani, O. (2016). Arabic Text Categorization Using Support vector machine, Naïve Bayes and Neural Network. *GSTF Journal on Computing*, 5(1), 108-115.

[12] Amasyalı, F. M., & Diri, B. (2006). Automatic Turkish Text Categorization in Terms of Author, Genre and Gender. *Natural Language Processing and Information Systems*. Berlin.

[13] Cutler, A., Stevens, J. R., & Cutler, D. R. (2011). Random Forests. In *Ensemble Machine Learning*, 157-176, Springer.

[14] Duan, K., Keerthi, S. S., Chu, W., Shevade, K. S., & Poo, A. N. (2003). Multi-Category Classification by Soft-Max Combination of Binary Classifiers. *Springer*. Berlin.

[15] Yıldırım, S., & Yıldız, T. (2018). A comparative analysis of text classification for Turkish language. *Pamukkale University Journal of Engineering Sciences*, 24(5), 879-886.

[16] Güran, A., Akyokuş, S., Bayazıt, N. G., & Gürbüz, M. Z. (2009). Turkish Text Categorization Using N-Gram Words. *International Symposium on Innovations in Intelligent Systems and Applicaitons*. Trabzon.

[17] Uysal, A. K., & Gunal, S. (2014). The Impact of Preprocessing on Text Classification. *Information Processing and Management*, 50(1), 104-112.

[18] Tapsai, C., Haruechaiyasak, C., & Meesad, P. (2016). TLS-ART: Thai Language Segmentation by Automatic Ranking Trie. T*he 9th International Conference Autonomous Systems*. Millor.

[19] Akın, A. A., & Akın, M. D. (2007). Zemberek, an open source NLP framework for Turkic Languages. *Structure*, 10, 1-5.

[20] Kayabaş, A., Schmid, H., Topcu, A. E., & Kılıç, Ö. (2019). TRMOR: a finite-state-based morphological analyzer for Turkish. *Turkish Journal of Electrical Engineering & Computer Sciences*, 3837-3851.

[21] Oflazer, K. (2014). Turkish and its challenges for language processing. *Language Resources and Evaluation*, 48(4), 639-653.

[22] Aksoy, A. (2016). Github. Retrieved 10 15, 2019, from https://github.com/ahmetax/trstop.

[23] pandas. (n.d.). Retrieved 09 23, 2019, from https://pandas.pydata.org.

[24] matplotlib. (n.d.). Retrieved 09 24, 2019, from https://matplotlib.org/.

[25] SciPy. (2017, 06). Retrieved 09 24, 2019, from https://docs.scipy.org/doc/numpy-1.13.0/user/whatisnumpy.html.

[26] Pedregosa, F. (2011). Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* (12), 2825-2830.

[27] Akın, A. A. (2014). *GitHub*. Retrieved 08 01, 2019, from https://github.com/ahmetaa/zemberek-nlp.

[28] Faggella, D. (2019, 02). *Emerj*. Retrieved 07 16, 2019, from https://emerj.com/ai-glossary-terms/what-is-machine-learning/.

[29] Leopord, H., Cheruiyot, W. K., & Stephen, K. (2016). A Survey and Analysis on Classification and Regression Data Mining Techniques for Diseases Outbreak Prediction in Datasets. *The International Journal of Engineering And Science*, 5(9), 01-11.

[30] Cheng, J., & Russell, G. (1999). Comparing Bayesian Network Classifiers. *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*. Stockholm.

[31] Prabhavathi, C., Vishali, N., Reddy, P. S., & Chandramouli, J. V. (2019). Machine Learning Model for Classifying L_Text Using NLP (Amazon Product Reviews). *International Research Journal of Computer Science*, 6(4), 161-178.

[32] Kibriya, A. M., Frank, E., Pfahringer, B., & Holmes, G. (2004). Multinomial Naive Bayes for Text Categorization Revisited. *Advances in Artificial Intelligence*. Cairns.

[33] Stigler, S. M. (1982). *Journal of the Royal Statistical Society*, 145, 250-258.

[34] Boser, B. E., Guyon, I. M., & Vapnik, V. N. (1992). A Training Algorithm for Optimal Margin Classier. *Proceedings of the fifth annual workshop on Computational learning theory*. New York.

[35] Nanda, M. A., Seminar, K. B., Nandika, D., & Maddu, A. (2018). A Comparison Study of Kernel Functions in the Support Vector Machine and Its Application for Termite Detection. Information, 9(5).

[36] Korkmaz, M., Güney, S., & Yiğiter, Ş. Y. (2012). The Importance of Logistic Regression Implementations In The Turkish Livestock Sector And Logistic Regression Implementations/Fields. *Harran Üniversitesi Ziraat Fakültesi Dergisi*, 2(16), 25-36.

[37] *Ronny Restrepo.* (2017). Retrieved 10 21, 2019, from http://ronny.rest/blog/post_2017_08_10_sigmoid/.

[38] *Global Software Support*. (2018, 02 23). Retrieved 07 20, 2019, from https://www.globalsoftwaresupport.com/random-forest-classifier-bagging-machine-learning/.

[39] Maini, V. (2017, 08). Retrieved 07 16, 2019, from https://medium.com/machine-learning-for-humans/unsupervised-learning-f45587588294.

[40] Hamza, A. (2018, 07). *Medium*. Retrieved 07 16, 2019, from https://medium.com/the-21st-century/machine-learning-a-strategy-to-learn-and-understand-chapter-3-9daaad4afc55.

[41] Ikonomakis, E. K., Tampakas, V., & Kotsiantis, S. (2005). Text Classification Using Machine Learning Techniques. *WSEAS Transactions on Computers,* 4(8), 966-974.

[42] Jindal, R., Malhotra, R., & Jain, A. (2015). Techniques for text classification: Literature review and current trends. *Webology*, 12(2).

[43] Schneider, C. (2016). The biggest data challenges that you might not even know you have. *IBM Watson*.

[44] M., K., Woonna, S., & Giri, P. (2016). Sentiment Analysis of Twitter Data. *International Journal of Innovations in Engineering and Technology*, 264-273.

[45] Sandeep, K., & Kaur Khiva, N. (2016). Online news classification using Deep Learning Technique. *International Research Journal of Engineering and Technology*, 558-563.

[46] Haddi, E., Liu, X., & Shi, Y. (2013). The Role of Text Pre-processing in Sentiment Analysis. *Procedia Computer Science*, 26-32.

[47] Hoonlor, A. (2011). *Sequential Patterns and Temporal Patterns for Text Mining*. New York: Rensselaer Polytechnic Institute.

[48] *Spring*. (n.d). Retrieved 5 12, 2019, from https://spring.io/projects/spring-boot.

[49] Kaviani, P., & Dhotre, S. (2017). Short Survey on Naive Bayes Algorithm. *International Journal of Advance Engineering and Research Development, 4*(11).

[50] Evgeniou, T., & Pontil, M. (2001). Support Vector Machines: Theory and Applications. *Machine Learning and Its Applications: Advanced Lectures* 249-257.

[51] Yadav, J., & Sharma, M. (2013). A Review of K-mean Algorithm. *International Journal of Engineering Trends and Technology, 4*(7), 2972-2976.

[52] Bishop, C. M. (2006). *In Pattern Recognition and Machine Learning* (331-333). Berlin: Springer.

[53] Peng, Joanne & Lee, Kuk & Ingersoll, Gary. (2002). An Introduction to Logistic Regression Analysis and Reporting. *Journal of Educational Research*, 96. 3-14.

[54] Aizawa, A. (2003). An Information-Theoretic Perspective of TF-IDF Measures. *Information Processing & Management, 39*(1), 45-65.

[55] Al-Harbi, S., Almuhareb, A., Al-Thubaity, A., Khorsheed, M. S., & Al-Rajeh, A. (2008). Automatic Arabic Text Classification.: *Proceedings of The 9th International Conference on the Statistical Analysis of Textual Data*. France.

[56] Sepp Hochreiter and Jürgen Schmidhuber (1997). Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780.

[57] Yangsen, Z., Jia, Z., Gaijuan, H., & Ruoyu, C. (2019). A Text Sentiment Classification Modeling Method Based on Coordinated CNN-LSTM-Attention Model. *Chinese Journal of Electronics, 28*, 120-126.

[58] Guimarães, R. G., Rosa, R., Gaetano, D. D., Rodríguez, D., & Bressan, G. (2017). Age Groups Classification In Social Network Using Deep Learning. *IEEE Access, 5*, 10805-10816.

[59] Zhang, X., Zhao, J., & LeCun, Y. (2016). Character-level Convolutional Networks for Text Classification. *arXiv e-prints*, 1609-1718.

[60] Parwez, A., Abulaish, M., & Jahiruddin. (2019). Multi-Label Classification of Microblogging Texts Using Convolution Neural Network. *IEEE Access, 7*, 68678-68691.

[61] *Keras*. (n.d). Retrieved 3 12, 2019, from keras.io: https://keras.io

[62] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 1929-1958.

[63] Olah, C. (2015, 08 27). *Reskilling* IT. Retrieved 12 09, 2019, from http://colah.github.io/posts/2015-08-Understanding-LSTMs.

# APPENDICES

**Appendix A:** List of Stop Words

**Appendix B:** List of Custom Stop Words

**Appendix C:** LSTM Model Training Code

**Appendix D:** K-Means Clustering with Sckit-learn

**Appendix E:** Dataset Analysis Code

**Appendix F:** Training Code of Traditional ML Algorithms

# Appendix A - List of Stop Words

"a","acaba","altı","altmış","ama","ancak","arada","artık","asla","aslında","ayrıca","az","bana","bazen","bazı","bazıları","belki","ben","benden","beni","benim","beri","beş","bile","bilhassa","bin","bir","biraz","birçoğu","birçok","biri","birisi","birkaç","birşey","biz","bizden","bize","bizi","bizim","böyle","böylece","bu","buna","bunda","bundan","bunlar","bunları","bunların","bunu","bunun","burada","bütün","çoğu","çoğunu","çok","çünkü","da","daha","dahi","dan","de","defa","değil","diğer","diğeri","diğerleri","diye","doksan","dokuz","dolayı","dolayısıyla","dört","e","edecek","eden","ederek","edilecek","ediliyor","edilmesi","ediyor","eğer","elbette","elli","en","etmesi","etti","ettiği","ettiğini","fakat","falan","filan","gene","gereği","gerek","gibi","göre","hala","halde","halen","hangi","hangisi","hani","hatta","hem","henüz","hep","hepsi","her","herhangi","herkes","herkese","herkesi","herkesin","hiç","hiçbir","hiçbiri","i","ı","için","içinde","iki","ile","ilgili","ise","işte","itibaren","itibariyle","kaç","kadar","karşın","kendi","kendilerine","kendine","kendini","kendisi","kendisine","kendisini",
"kez","ki","kim","kime","kimi","kimin","kimisi","kimse","kırk","madem","mi","mı",
"milyar","milyon","mu","mü","nasıl","ne","neden","nedenle","nerde","nerede","nereye","neyse","niçin","nin","nın","niye","nun","nün","o","öbür","olan","olarak","oldu","olduğu","olduğunu","olduklarını","olmadı","olmadığı","olmak","olması","olmayan","olmaz","olsa","olsun","olup","olur","olur","olursa","oluyor","on","ön","ona","önce","ondan","onlar","onlara","onlardan","onları","onların","onu","onun","orada","öte","ötürü","otuz","öyle","oysa","pek","rağmen","sana","sanki","şayet","şekilde","sekiz","seksen","sen","senden","seni","senin","şey","şeyden","şeye","şeyi","şeyler","şimdi","siz","sizden","sizden","size","sizi","sizi","sizin","sizin","sonra",
"şöyle","şu","şuna","şunları","şunu","ta","tabii","tam","tamam","tamamen","tarafından","trilyon","tüm","tümü","u","ü","üç","un","ün","üzere","var","vardı","ve","veya",
"ya","yani","yapacak","yapılan","yapılması","yapıyor","yapmak","yaptı","yaptığı","yaptığını","yaptıkları","ye","yedi","yerine","yetmiş","yi","yı","yine","yirmi","yoksa","yu","yüz","zaten","zira"

# Appendix B - List of Custom Stop Words

"mahalle","mh","mah","sokak","cadde","no","cep","tel","faks","fax",
"cad","sok","sk","te","iç","kap","bulvar","il","ilçe","gerek","arz","etmek","sayın",
"başvuru", "istinaden", "null", "saat", "civar", "ara", "ora", "bura", "kişi", "başlamak",
"görev", "sıkıntı", "yaşamak", "binmek", "ad", "soy", "taraf", "acilen", "müdahale",
"çöz", "bulunmak", "eski","meydan", "gelmek", "bilgi", "numara", "birim", "vermek",
"yarmak", "söz","iyi", "günlemek", "sayın", "başkan", "yetkili", "tarih", "mağdur",
"şikayet", "vatandaş", "nol","yeni", "anmak", "zor", "kalmak",
"demek","almak","bina","gitmek","patlak","ivedilikle","mağdur","temiz","yolmak","
konu","durum","ev","geçmek","kontrol","ivedi","nol","rica","gün","mevcut","park",
"kullanmak","site","beklemek","bey","yok","lütfen","talep","istemek","şikâyet","bel
ediye","büyükşehir","mağduriyet","gidermek"

# Appendix C – LSTM Model Training Code

```
MAX_NB_WORDS = 50000
EMBEDDING_DIM = 100
def buildModel():
    model = Sequential()
    model.add(Embedding(MAX_NB_WORDS, EMBEDDING_DIM,
input_length=X.shape[1]))
    model.add(SpatialDropout1D(0.2))
    model.add(LSTM(100, dropout=0.2, recurrent_dropout=0.2))
    model.add(Dense(14, activation='softmax'))
    model.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])
    print(model.summary())
    return model

model = buildModel()
model.fit(X_train, Y_train,
                    epochs=epochs,
                    batch_size=batch_size,
                    validation_split=0.1,
                    callbacks=[EarlyStopping(monitor='val_loss',
patience=3, min_delta=0.0001)])
```

# Appendix D – K-Means Algorithm with Sckit-learn

```
vectorizer = TfidfVectorizer()
X = vectorizer.fit_transform(corpus)

true_k = 14
model = KMeans(n_clusters=true_k, init='k-means++', max_iter=100,
n_init=1)
model.fit(X)

order_centroids = model.cluster_centers_.argsort()[:, ::-1]
terms = vectorizer.get_feature_names()

for i in range(true_k):
    print("Cluster %d:" % i),
    for ind in order_centroids[i, :10]:
        print('%s' % terms[ind])

vectorizedTextData = vectorizer.transform([topicTitle])
predicted = model.predict(vectorizedTextData)
```

# Appendix E – Dataset Analysis Code

```
import pandas as pd
import json

dataset = pd.read_json(path_or_buf=jsonFilePath',encoding="utf8")

print("##Word Count İstatistikleri")
print(dataset.word_count.describe())

freq = pd.Series('
'.join(dataset['text']).split()).value_counts()[:21]
print(freq)
freq.to_excel("TextCommon21unigramword.xlsx")

for index, row in dataset.iterrows():
    corpus.append(row['text'])

from PIL import Image
from wordcloud import WordCloud, ImageColorGenerator
import matplotlib.pyplot as plt

wordcloud = WordCloud(
    background_color='white',
    max_words=100,
    width=800, height=400,
    max_font_size=50,
    random_state=42
).generate(str(corpus))


fig = plt.figure(1)
plt.imshow(wordcloud)
plt.axis('off')
plt.show()
fig.savefig("yeniWordCloud.png", dpi=900)
```

# Appendix F – Training Code of Traditional ML Algorithms

```
import json
import pandas as pd
import numpy as np
import time
import matplotlib.pyplot as plt


with open('CorpusJsonFileLocation, encoding='utf-8') as corpus:
    data = json.load(corpus)
df = json_normalize(data)

fig = plt.figure(figsize=(8, 6))
df.groupby('topic').text.count().plot.bar(ylim=0)
plt.show()

from sklearn.model_selection import train_test_split
from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfTransformer
from sklearn.naive_bayes import MultinomialNB

X_train, X_test, y_train, y_test, indices_train, indices_test =
train_test_split(features, labels, df.index,
train_size=0.75,test_size=0.25, random_state=0)


count_vect = CountVectorizer()
X_train_counts = count_vect.fit_transform(X_train)
tfidf_transformer = TfidfTransformer()
X_train_tfidf = tfidf_transformer.fit_transform(X_train_counts)

naiveBayes = MultinomialNB().fit(X_train_tfidf, y_train)

model = SVC(kernel='linear')
model.fit(X_train, y_train)

model = RandomForestClassifier(n_estimators=200, max_depth=3,
random_state=0)
model.fit(X_train, y_train)

model = LogisticRegression(random_state=0)
model.fit(X_train, y_train)
```

# CURRICULUM VITAE

## PERSONAL INFORMATION

**Name Surname** : Ahmed Enis ERKAYA
**Date of Birth** : 24.09.1995
**Phone** : +90 554 415 7979
**E-mail** : aeniserkaya@gmail.com

## EDUCATION

**High School** : Nefise Andiçen High School - 2013
**Bachelor** : Dumlupınar University - 2017
**Master Degree** : Ankara Yıldırım Beyazıt University - 2019

## WORK EXPERIENCE

**Software Engineer:** TÜBİTAK BİLGEM Software Technologies Research Institute 2018 - Present

**Software Engineer:** Emsal Bilişim 2017 - 2018

## TOPICS OF INTEREST

- Software Engineering
- Machine Learning
- Mobile Applications
- Cloud Computing

## PROCEEDING PAPER

Erkaya, A. E., & Topcu, A. E. (2019). “Text Classification Using Machine Learning Techniques for Citiziens' Communication”. *International Symposium on Academic Studies in Science, Engineering and Architecture Sciences,* (239-245). Ankara