T.C.

BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL OF EDUCATION

ARTIFICIAL INTELLIGENCE HEAD OF THE DEPARTMENT

ENHANCING CALL CENTER EFFICIENCY THROUGH MAIL PARSING USING NATURAL LANGUAGE PROCESSING AND ROBOTIC PROCESS AUTOMATION

MASTER'S THESIS

REFİH CAN

İSTANBUL 2024

T.C.

BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL OF EDUCATION

ARTIFICIAL INTELLIGENCE HEAD OF THE DEPARTMENT

ENHANCING CALL CENTER EFFICIENCY THROUGH MAIL PARSING USING NATURAL LANGUAGE PROCESSING AND ROBOTIC PROCESS AUTOMATION

MASTER'S THESIS

REFİH CAN

THESIS ADVISOR
Prof. Dr. Süreyya AKYÜZ

İSTANBUL 2024

T.C.

BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL

07/06/2024

MASTER THESIS APPROVAL FORM

Program Name:	**ARTIFICIAL INTELLIGENCE**
Student's Name and Surname:	**Refih CAN**
Name Of The Thesis:	**Enhancing Call Center Efficiency Through Mail Parsing Using Natural Language Processing and Robotic Process Automation**
Thesis Defense Date:	**07.06.2024**

This thesis has been approved by the Graduate School which has fulfilled the necessary conditions as Master thesis.

Assoc. Prof. Yücel Batu SALMAN

Director of Graduate School

This thesis was read by us, quality and content as a Master's thesis has been seen and accepted as sufficient.

	Title, Name	**Institution**	**Signature**
Thesis Advisor:	**Süreyya Akyüz**	**BAU**	
2nd Member	**Dr. Öğr. Üyesi Tarkan Aydın**	**BAU**	
3rd Member (Outside Institution)	**Prof. Dr. Birsen Eygi Erdoğan**	**Marmara Üniversitesi**	

Bu tezdeki tüm bilgilerin akademik kurallara ve etik ilkelere uygun olarak elde edildiğini ve sunulduğunu; ayrıca bu kuralların ve ilkelerin gerektirdiği şekilde, bu çalışmadan kaynaklanmayan bütün atıfları yaptığımı beyan ederim.

Ad, Soyad :

İmza :

ABSTRACT

ENHANCING CALL CENTER EFFICIENCY THROUGH MAIL PARSING USING NATURAL LANGUAGE PROCESSING AND ROBOTIC PROCESS AUTOMATION

Can, Refih
Master's Program in Artificial Intelligence
Supervisor: Prof. Dr. Süreyya Akyüz

June 2024, 42 pages

The call center industry is undergoing a transformative shift towards automation and improved operational efficiency. Traditionally, call centers have relied on manual handling of incoming mails, which is time-consuming and error-prone. This thesis explores the integration of Natural Language Processing (NLP) and Robotic Process Automation (RPA) technologies to streamline the process of mail parsing within call centers. This research begins by analyzing the current challenges faced by call centers in managing incoming emails and highlights the potential benefits of automating mail parsing. NLP techniques are employed to extract valuable information from emails, such as customer requests, issues, and sentiment analysis, while RPA bots are utilized to classify, automate the processes according to the classification, and route emails to the appropriate agents. After the creation of the automation, comparisons of NLP techniques, and their performances on different datasets are examined.

Keywords – Intelligent Process Automation, Robotic Process Automation, Natural Language Processing, Machine Learning, Mail Parsing

DEDICATION

I would like to express my endless and sincere gratitude during this study to my advisor Prof. Dr. Süreyya Akyüz, whose valuable knowledge and experience I benefited from, jury member Dr. Lecturer Member Tarkan Aydin and Prof. Dr. Birsen Eygi Erdoğan, all my Assistt colleagues, who supported me in my work, my family, who brought me to this day, and my lovely life partner who showed me the greatest strength and faith in this thesis work. I dedicate this thesis to my wife, Jennifer Brooke Lanz.

THANKS

I would like to express my endless gratitude to my teacher, Professor Doctor Süreyya Akyüz, who showed interest and support in the planning, research, execution, and formation of this thesis study, benefited from his vast knowledge and experience, and shaped my work in the light of scientific foundations with his guidance and information.

CONTENT

TABLES LIST

FIGURES LIST

EQUATION LIST

ABBREVIATIONS LIST

RPA	Robotic Process Automation
NLP	Natural Language Processing
ML	Machine Learning
SVM	Support Vector Machine
K-NN	K Nearest Neighbor
NER	Named Entity Recognition
BOW	Bag of Words
TF	Term Frequency
IDF	Inverse Document Frequency
MLE	Maximum Likelihood Estimation

CHAPTER 1

Introduction

In the age of Industry 4.0, which is defined by the seamless integration of digital technology into all aspects of business operations, the call center industry is at the forefront of transformation. As organizations endeavor to navigate the complexities of customer service in an increasingly interconnected world, the convergence of Natural Language Processing (NLP) and Robotic Process Automation (RPA) emerges as a light of innovation, promising unparalleled efficiency and effectiveness (Hirschberg & Manning, 2015).

At the core of this technological revolution lies NLP, a branch of artificial intelligence that enables machines to comprehend and interpret human language with remarkable accuracy (Hirschberg & Manning, 2015). By harnessing the power of NLP, call centers can unlock valuable insights from the vast volumes of unstructured text data flowing through their systems, empowering them to extract meaning, categorize inquiries, and tailor responses with unprecedented precision.

Yet, NLP is but one piece of the puzzle. In tandem with NLP, RPA emerges as a formidable force, offering the capability to automate repetitive tasks and streamline workflow processes. By deploying software robots to handle routine activities such as data entry, ticket generation, and email routing, call centers can liberate human agents from mundane tasks, allowing them to focus their energies on value-added activities that require human intuition and empathy (Madakam et al., 2019).

Automating email parsing processes is crucial for businesses, offering transformative benefits across various organizational areas. The trend of relying on

manual labor or costly automated solutions underscores the urgent need for more efficient and cost-effective alternatives. By leveraging Natural Language Processing (NLP) and Robotic Process Automation (RPA), organizations can significantly enhance efficiency, reduce costs, and improve customer experience. Moreover, the integration of NLP and RPA technology presents opportunities for innovation and competitiveness in the evolving landscape of customer service operations. Despite the clear benefits and the prevalent need, there is a noticeable lack of scientific interest in automating email parsing processes with NLP and RPA. This gap in research highlights the importance of exploring this area to fully realize its potential for organizational success and technological advancement.

In this thesis, an in-depth study on integrating NLP and RPA to automate email parsing processes, particularly for Turkish emails, was conducted. The thesis comprises five chapters, each addressing key aspects of our research. First, the problem description was refined, highlighting the unique contributions of our work to email automation. Then, an in-depth review of NLP was provided, explaining key concepts and application steps for Turkish emails. Next, the functionality of 12 machine learning algorithms was examined, evaluating their suitability for Turkish email classification. The following chapter explains how RPA can simplify and automate email processing. Finally, a rigorous benchmarking of the 12 machine learning algorithms was conducted, culminating in a comprehensive SWOT analysis summarizing the strengths, weaknesses, opportunities, and threats of our research. This multifaceted approach provides a comprehensive overview of automating email parsing processes, offering valuable insights for researchers and practitioners.

CHAPTER 2

Literature Survey

In the realm of artificial intelligence, NLP and Text Classification stand out as highly researched domains. In the context of an email classification project, a comprehensive literature review was undertaken to explore these areas, delving into similar studies for thorough examination, and the most important studies in terms of scope are listed. Dealing with text data in email bodies presents a challenge in selecting attributes to determine context and meaning (Fong et al., 2008). Feature extraction methods are actively used for text classification, but the multitude of possibilities poses difficulties, with some attributes proving more useful than others depending on email classes (Ayodele et al., 2007). Adaptive classification is necessary to address multiple email formats resulting from the lack of standardization (Carmona-Cejudo et al., 2011). While challenges persist in extracting attributes to relate text sections, achieving accuracy is feasible, albeit dependent on attribute selection (Chan et al., 2004). The large attribute sample space, especially with simple bag-of-words approaches, can pose problems for classification algorithms (Ayodele et al., 2007). Removal of redundant attributes is crucial for effective information extraction, as the performance of email classification algorithms hinges on their ability to extract relevant attributes (Wang et al., 2006). Mainstream algorithms for email classification are explored, underscoring the importance of attribute selection for efficient and accurate classification. The RIPPER algorithm (Provost, 1999) and Naïve Bayes classification (Haiyi & Li, 2007) are common in automatic email filtering. RIPPER's rule-based framework, though fast, requires strict keyword extraction rules, potentially leading to mix-ups. Naïve Bayes relies on statistical analysis, efficiently processing

emails but may struggle with large feature vectors. Nearest neighbor classification, as explored in a study (Weijie et al., 2012), uses mutual information for feature selection, treating feature vectors in n-dimensional space for effective matching. Neural networks, though sophisticated, are still evolving for intelligent email filtering, utilizing backpropagation for training and generalization (Arevian, 2007). Advances in intelligent email filtering include preprocessing techniques to gain statistics, eliminating the need for stop-word lists, and exploring machine learning methods for automation tasks like email reply prediction (Yang and Kwok, 2012). Segmenting data extraction into separate layers can enhance scalability (Dredze, 2009), while semantic-based approaches show promise in improving email communication (Beseiso et al., 2012). Statistical methods offer language-independent attribute extraction, but research lacks in NLP for feature selection and implementation of current new ML algorithms, which could improve the classification of unclassified emails (Yang & Kwok, 2012). In NLP algorithms, it is crucial to initially cleanse the text by removing stop words like conjunctions, prepositions, pronouns, and punctuation marks. Text classification employs stemming, aimed at deriving the fundamental roots of words, and lemmatization, which simplifies words by considering their morphology. While some literature (Çağataylı & Çelebi, 2015; Deniz & Kiziloz, 2017) suggests that stemming minimally impacts subject classification in Turkish texts, lemmatization algorithms demonstrate a beneficial effect on information extraction (Öztürkmenoğlu & Alpkoçak, 2012). Furthermore, studies focus on enhancing existing algorithms for this purpose (Tahiroğlu, 2021), either through comparison or integration (Salur et al., 2021; Yıldırım & Yıldız, 2018). Text classification, a frequently explored research area, particularly emphasizes the development of machine learning-based algorithms (Khan, 2010; Kadhim, 2019; Küçük & Arıcı, 2018). While research predominantly

focuses on English texts (Dharmadhikari et al., 2011; Kandimalla et al., 2021), investigations also extend to Turkish texts (Gurcan, 2018; Küçük & Arıcı, 2018). Text classification entails either assigning texts to predefined class sets or dividing them into groups without predefined classes. Typically, supervised machine learning algorithms are employed for the former, as observed in this study. Notably, studies on text classification in Turkish texts have gained significance, employing various techniques such as deep learning, Hidden Dirichlet Discrimination, and hybrid approaches, which integrate methods like Naïve Bayes, Support Vector Machines, and J48 (Kilimci & Akyokus, 2019; Kilimci & Akyokus, 2018; Aydin & Hallaç, 2021; Erşahin et al., 2019). Though numerous studies focus on text and email classification, only a few incorporate NLP techniques, with just one specifically tailored for the Turkish language (Sel & Hanbay, 2019). In this particular study, emails are clustered using k-means, an unsupervised method.

Willcocks (2017) asserted that robotic process automation excels at executing repetitive tasks swiftly and accurately, relieving humans to engage in higher-value activities requiring judgment, customer interaction, and emotional intelligence. Report Bataller (2017) on US9555544B2 described robotic process automation (RPA) as a technology facilitating the automation of repetitive and manually intensive tasks, wherein a computer system or robot mimics human actions to perform computer-based tasks. Sharma (2019) on US10354225B2 emphasized the necessity of robotic process automation in assisting users and reducing their workload, advocating for teaching processes to alleviate this burden. He highlighted that in RPA technology, robots are granted authority to perform tasks akin to humans, thereby enabling the automation of various computer-based business processes. Bataller (2019) in EP3215900B1 conveyed that integrating robotic process automation with existing applications at a

coding level is unnecessary. United States patent document US2018197123 (2018) pertained to a platform comprising development and operational databases stored in memory, alongside components such as a development interface, control interface, and runner component. Finally, US patent document US20170352041 (2017) focused on robotic process automation (RPA) within supply chain management (SCM) operations. In this invention, exemplified embodiments depict RPA systems incorporating IoT devices and a server within a subnet. The innovation leverages AI-driven procurement processes and processing logic to ensure secure communication concerning SCM transactions. Consequently, the automation bot for SCM processes minimizes processing durations while enhancing operational efficiency. United States patent document No. US2018197123 (2018) employs the outlined controller execution model to interconnect systems, computer program products, and methods for automated implementation via robotic processes. The invention is designed to electronically receive user data. Within the invention, HVD bots coexist with a computer device linked to the bot, associated with the controller-hosted virtual desktop hub, establishing connectivity with the bot to the hosted virtual desktop upon a communication request with the controller. According to Çalışkan (2020), based on the outcomes of their evaluative survey, all participants (100%) affirmed the judicious decision to adopt robotic process automation technology, with 61.9% attaining the anticipated benefits and 23.8% expressing otherwise. During the industrial era, advancements in RPA and AI technologies have significantly enhanced operational efficiency, particularly in sectors like banking, insurance, and call center services, where data generation is prolific. It was posited that these technologies not only aim to safeguard but also expand customer networks, offering a transformative opportunity. As robotic process automation systems evolve, they are poised to tackle

increasingly intricate tasks with the aid of artificial intelligence, heralding the development of a more cognitively adept digital workforce. The assertion was made that artificial intelligence, with its promise of expediting processes, enhancing convenience, and optimizing costs, stands at the forefront of organizational metamorphosis. Erdoğan (2020) stressed the criticality of robust testing procedures in RPA implementations, given that robots operate autonomously, suggesting that tests should be conducted under human supervision. Uğurlu (2019) advocated for RPA technology, highlighting its potential to streamline collaborative efforts across diverse business domains while enabling uninterrupted, round-the-clock operation of end-to-end processes. Damar (2022) contended that the process of adapting established business procedures faces challenges due to the evolving nature of the systems in which robots operate within processes governed by robotic process automation. Murat Akyıldız (2007) emphasized the significance of understanding the labor and resources necessary for a given task in order to accurately estimate the associated costs in terms of labor days, underscoring the importance of comprehending the scope and scale of the project at hand for effective management. Borandağ et al. (2013) critiqued the method of estimating labor day costs solely based on the number of process lines to be developed, noting its limitations when considering the experience level of the developer, suggesting that it should not be relied upon as the sole estimation method. Gür et al. (2019) posited that artificial intelligence technologies find applications within Human Resources departments, encompassing functions such as recruitment, business process monitoring, identification of training needs, and performance evaluation and management. Braidotti (1997), Ferrando (2014), Pitts and Dinerstein (2017), Romero et al. (2016), Rose et al. (2016) conducted in-depth analyses of studies on Robotic Process Automation (RPA) amidst the evolving landscape of Industry 4.0

technologies, sharing their research findings through various applications. Meanwhile, Sotala and Yampolskiy (2013) explored the realm of robotic technology and its implications, offering insights into potential future directions for these technologies. Albayrak (2020) contended that NLP technology embodies the capability to comprehend human speech, acknowledging the complexities inherent in human language such as dialects, slang, and socio-cultural influences, which pose challenges for the development of NLP systems. Similarly, Noyan (2019) posited that NLP technology has progressed in tandem with text mining, facilitating the processing of vast amounts of data and the extraction of meaningful insights. Moreover, Maček (2020), Figueiredo (2021), Golizadeh (2020), Santos (2020), Urbahs (2019), and Duan (2019) scrutinized case studies involving the integration of call centers with robotic systems, evaluating the efficacy of innovative approaches. Finally, Priyadarshi (2022), Zhang (2019), Huettinger (2020), and Lyne (2018) delved into the intricate relationships between innovation and RPA, offering detailed examinations of their interplay.

In contrast to previous works that employed solely the k-means algorithm and a limited selection of RPA platforms, our study takes a more comprehensive approach. 12 different machine learning algorithms are utilized to analyze a Turkish dataset, providing benchmark performances for each. Additionally, another advanced RPA platform is incorporated to enhance the overall process. This expanded methodology allows for a more robust comparison and deeper insights into the effectiveness of various machine learning techniques in conjunction with RPA technology.

CHAPTER 3

Methodology

3.1. Problem Statement

In the nascent stages of Robotic Process Automation (RPA), the primary objective centered on automating the repetitive branches within business processes, typically handled by employees due to system API inaccessibility. These branches, characterized by minimal exceptions, confined cognitive scope, and susceptibility to human error, were identified as prime candidates for RPA intervention. However, the horizontal segmentation of processes, aimed at achieving full automation of ideal branches, proved to be of limited applicability, as such branches were rarely encountered (cf. left side of Figure 1). Conversely, the concept of vertical segmentation emerged as a more pragmatic approach (cf. right side of Figure 1).

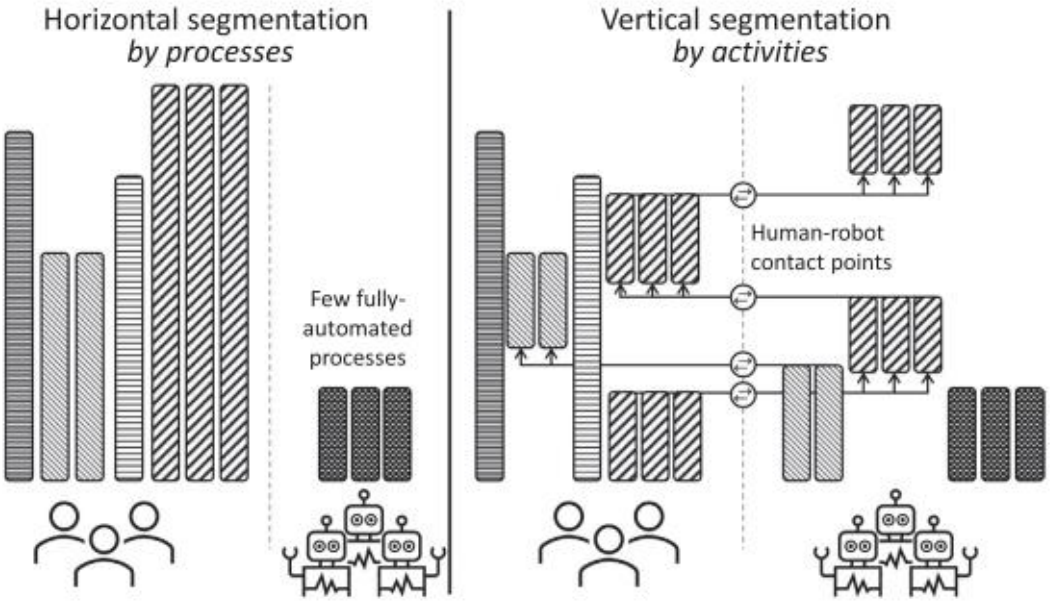

Figure 1. Workload assignments to humans and robots in both horizontal and vertical segmentation approaches (Ruiz R. C. et al., 2022).

Here, certain activities were automated while others remained under human purview, giving rise to a hybrid collaboration paradigm between robots and humans. This collaboration, integrating the "human-in-the-loop" concept, can be viewed as either a human-centric process augmented by robotic activities or vice versa. Various collaborative models, such as asynchronous and synchronous approaches, have been employed, with asynchronous collaboration being the more prevalent choice. In this setup, independent task queues are maintained for both robots and humans, facilitating the efficient execution of tasks. Asynchronous collaboration finds particular utility in scenarios where robots preprocess data before human intervention, allowing for swift cognitive actions by humans. However, despite the introduction of automation, cognitive tasks handled by humans still comprise a significant workload. Currently, in most organizations, email classification tasks are carried out manually by employees, with only a small fraction employing expensive fully automated solutions to handle these cognitive tasks.

3.2. Contribution

Our article represents a significant advancement by merging NLP with RPA, marking only the second instance documented in the literature. While the initial endeavor focused on invoice processing, our thesis pioneers this integration within the call center industry. Moreover, although numerous studies utilize NLP for email classification, they predominantly center on English emails. Our contribution extends this research landscape by introducing the second paper dedicated to classifying Turkish emails. In contrast to the previous work employing solely the k-means algorithm, our study employs 12 novel machine learning algorithms, 5 of which are newest, meticulously benchmarking their performances. Beyond academic

contributions, our work presents a compelling commercial opportunity, poised to be leveraged within the sector.

3.3. Solution

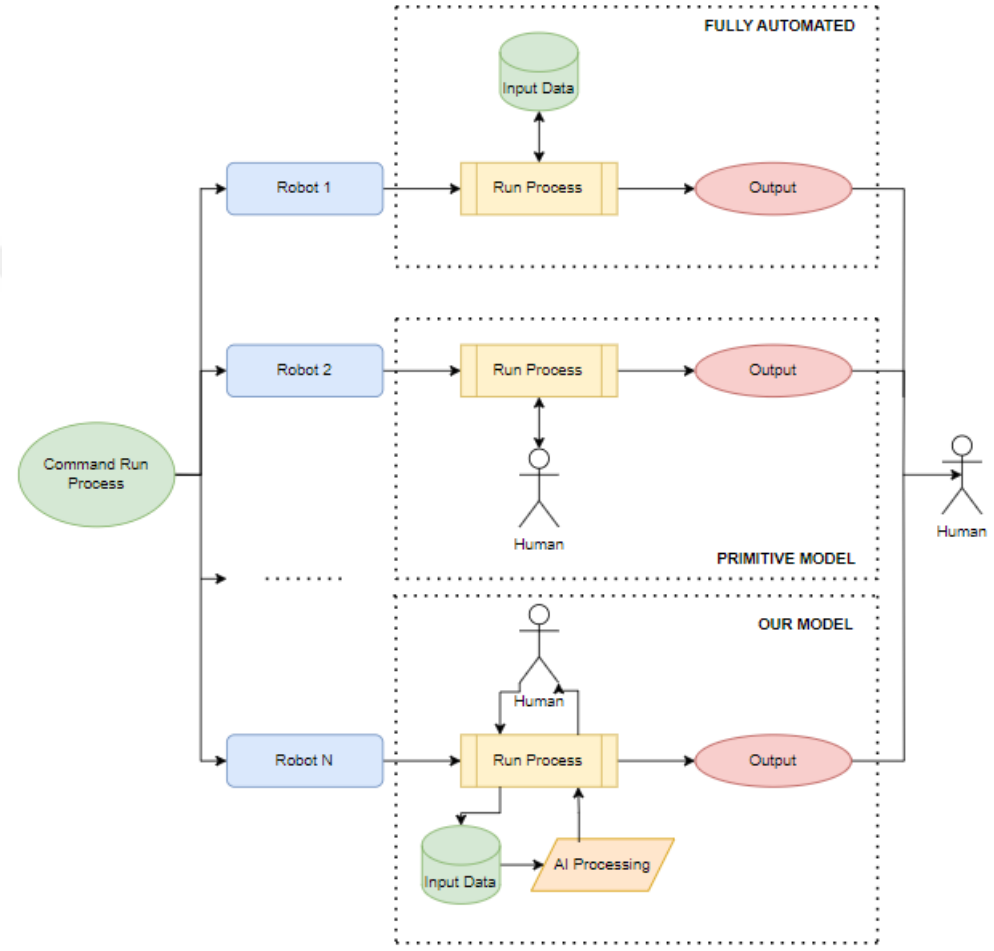

Figure 2. Email classification and processing in a primitive model, full automated model, and our model.

In response to the challenges posed by manual email classification processes and the limitations of fully automated systems, our solution introduces an innovative approach that combines real-time communication with humans and advanced automation techniques. At the heart of our solution lies a dynamic collaboration model, where humans and robots work synergistically to handle complexities and exceptions

efficiently. Leveraging NLP and ML algorithms, our system preprocesses email data, extracting key features and categorizing them accurately. Subsequently, RPA automates email processes based on their respective classifications, streamlining workflow and reducing manual intervention. This hybrid approach, characterized by a vertical segmentation strategy, enables seamless integration of AI-driven automation with human cognitive capabilities. By embracing this collaborative paradigm, our solution not only alleviates the cognitive burden on humans but also enhances the speed and accuracy of email classification tasks. Overall, our approach represents a significant advancement in email processing, offering a scalable and adaptable framework that optimizes efficiency while preserving the essential role of human judgment and oversight.

3.3.1. Data preparation. In preparation for training and testing ML algorithms for email classification at Turk Telekom, the challenge of insufficient data volume is encountered within the company's dataset, as the emails flowed in daily. To address this limitation, the dataset is augmented by randomly generating approximately 100,000 synthetic email samples, supplementing the original 1000 data. Subsequently, the combined dataset is partitioned using a 90-10 split ratio, allocating 90% of the data for training and the remaining 10% for testing. This approach ensured a representative distribution of data for both the training and evaluation phases. By employing this methodology, the objective is to strengthen the reliability of the trained models and simplify the process of benchmarking performance across 12 machine learning algorithms.

3.3.2. NLP. NLP is an area of artificial intelligence dedicated to facilitating communication between computers and humans using natural language. This field aims to equip computers with the ability to comprehend, interpret, and produce human language in a meaningful and practical manner. NLP includes a range of methods such as text analysis, sentiment analysis, language generation, and translation.

In this study, data preprocessing was initiated to prepare the textual data for clean and effective analysis. This involved several steps such as removing noise (e.g., punctuation, special characters), handling missing values, normalizing text by converting it to lowercase, and tokenizing sentences into words to prepare raw text data for further analysis. Following data preprocessing, feature extraction was conducted to convert the textual data into a format suitable for ML algorithms. Techniques such as Term Frequency-Inverse Document Frequency (TF-IDF) and CountVectorizer were employed to capture the semantic meaning and importance of the words. While CountVectorizer transforms a collection of text documents into a matrix of token counts, TF-IDF assists in determining a word's relevance inside a document in relation to a corpus.

Finally, 12 different ML algorithms were applied for classification, evaluating their performance to determine the most effective model for our model. These algorithms included but were not limited to, logistic regression, decision trees, support vector machines, and ensemble methods. The performance of each model was assessed based on metrics to identify the best-performing classifier for our dataset.

3.3.2.1. Data preprocessing. In the data preprocessing phase, the Zemberek library is utilized to enhance the handling of Turkish language data. The preprocessing steps carried out in our model are outlined as follows:

1. *Cleaning of Punctuation Marks:* Removing characters that are neither words nor whitespace from the text dataset is crucial. Punctuation marks, symbols, and other special characters, which do not add meaningful information for our model, should be excluded.
2. *Converting the Text to Lowercase Letters:* To prevent any problems and maintain uniformity in text processing, all text is transformed to lowercase. This ensures that “Data” and “data” are considered identical, enhancing the accuracy and reliability of our model.
3. *Text Normalization:* Numerous normalization procedures exist, but our approach primarily involved standardizing the formatting of data entries, such as dates and social security numbers, for consistency, and correcting spelling errors to ensure uniformity in word expression.
4. *Stemming:* The process of reducing words to their stem—their base or root—is called stemming. This is achieved by removing suffixes or prefixes from words, to reduce inflected words to their root form. The resulting stem may not always be a valid word, but it allows for grouping words with similar meanings.
5. *Cleaning of Stop Words:* The process of removing stop words involves eliminating common words that are considered to have little or no significance in the analysis of text data. These words, such as "hangi", "kendi", "neyse", "elbette" etc., occur frequently in language but typically do not contribute much to the overall meaning of the text. By removing stop words, the focus is

shifted to the more meaningful words in the text, which can improve the performance of text classification.

6. *Named Entity Recognition:* NER is a branch of NLP that deals with finding and categorizing named things in a text into pre-established groups, including names of people, places, dates, and numbers. Finding and labeling things in unstructured text with the appropriate categories is the aim of NER. Names of individuals and organizations are kept in the text in our study because they are essential to classification.

3.3.2.2. Feature extraction. Feature extraction is used in NLP to transform raw text data into a format that ML algorithms can understand and process effectively. In the study, both the CountVectorizer and TFIDF Vectorizer techniques are employed to extract the features on the email dataset. Initially, the performance of these two methods was evaluated individually and found that the TFIDF Vectorizer outperformed the CountVectorizer on the dataset. Encouraged by this result, the feature extraction stage is further refined by exploring the use of Bigram and Trigram TFIDF Vectorizers. By incorporating bi- and trigram features, our model aimed to capture more nuanced relationships between words and phrases in the text data.

The CountVectorizer is a feature extraction method used in the model to convert text documents into numerical vectors. This method belongs to the Bag of Words (BoW) family and is particularly useful for text classification and clustering tasks. CountVectorizer works by first tokenizing the text, meaning it is split into individual words or tokens. Then, CountVectorizer builds a vocabulary of all unique words (or tokens) in the corpus, assigning each word a unique index in the vocabulary.

For each document in the corpus, CountVectorizer counts the occurrences of each word in the vocabulary and constructs a numerical vector representing the document, where each element of the vector corresponds to the count of a particular word in the vocabulary. Since most documents only contain a small subset of the entire vocabulary, the resulting vectors are typically sparse, containing mostly zeros. CountVectorizer returns a sparse matrix representation of the document-term matrix to save memory and computational resources.

The TF-IDF Vectorizer is another feature extraction method used in the model to convert text documents into numerical vectors. TF-IDF stands for Term Frequency-Inverse Document Frequency. The TF-IDF Vectorizer operates by first tokenizing the text, splitting it into individual words or tokens. For each document in the corpus, the TF-IDF Vectorizer calculates the frequency of each term (word) in the document, similar to the CountVectorizer. In addition to the term frequency, TF-IDF also considers the inverse document frequency of each term, measuring how unique or important it is across the entire corpus. Words that appear frequently in many documents are penalized, while rare words that appear in only a few documents are given higher weights. The TF-IDF Vectorizer combines the term frequency and inverse document frequency to calculate the final weight of each term in the document, typically using the formula TF * IDF. The document-term matrix is returned by the TF-IDF Vectorizer as a sparse matrix representation, with each row denoting a document and each column denoting a distinct term in the vocabulary.

The TFIDF Bigram Vectorizer extends the TF-IDF technique to capture not only single words but also pairs of adjacent words (bigrams), providing richer contextual information for text. The TFIDF Bigram Vectorizer first tokenizes the input text documents into individual words or tokens. Instead of considering only single

words (unigrams), it also considers adjacent pairs of words (bigrams), generating a vocabulary of bigrams in addition to unigrams. For each document in the corpus, the TFIDF Bigram Vectorizer calculates the TF-IDF score for each unigram and bigram present in the document. By combining phrase frequency (TF) and inverse document frequency (IDF), the TF-IDF score calculates a term's relative relevance inside a document to the overall corpus. A term's frequency of occurrence in a document is measured by TF, but its rarity within the corpus is measured by IDF. Every document is created by the TFIDF Bigram Vectorizer with a numerical vector whose elements each indicate the TF-IDF score of a bigram or unigram in the text.

The TF-IDF Trigram Vectorizer works similarly to the TF-IDF Vectorizer but specifically focuses on trigrams, which are sequences of three consecutive words in a text. By using trigrams instead of individual words or bigrams, the TF-IDF Trigram Vectorizer captures more complex patterns and relationships within the text. The text is first tokenized into individual words or tokens, grouping words into sequences of three adjacent words. After tokenization, the term frequency (TF) of each trigram is calculated, representing the frequency of occurrence of each trigram within a document. The inverse document frequency (IDF) is calculated for each trigram across the entire corpus, measuring the importance of a trigram within the corpus by penalizing terms that appear frequently across documents. Rare trigrams receive higher IDF scores, indicating greater importance. Finally, the TF-IDF score is calculated for each trigram in each document by multiplying its TF by its IDF, giving a numerical representation of the significance of each trigram within a document relative to the entire corpus. Every document is represented as a vector with each dimension matching to a trigram and each dimension's value representing the TF-IDF

score of the corresponding trigram in the document once the TF-IDF scores for every trigram in every document have been calculated.

3.3.2.3. Classification with machine learning algorithms. The initial methods employed to classify emails often relied on rule-based approaches or simple keyword-matching techniques (Aery M. & Chakravarthy S., 2005; Tham, K.-T. et al., 2023). However, these methods are seen to be limited in their effectiveness, particularly when faced with the complexity and variability of natural language present in email communications. As a result, there has been a shift towards leveraging ML techniques for email classification (Khan A. et al., 2010). Leveraging ML algorithms for classification in NLP tasks, such as email classification, holds significant importance and offers various advantages. Firstly, ML algorithms enable automated processing and analysis of large volumes of textual data, which is essential for tasks like email classification where the volume of incoming messages can be overwhelming. By employing ML algorithms, NLP systems can efficiently categorize emails into different classes or labels based on their content, thus streamlining the process of email management and organization. Additionally, ML algorithms can adapt and learn from data patterns, allowing NLP models to continually improve their accuracy and effectiveness over time (Kadhim. A. I. et al., 2019). This adaptability is crucial in email classification tasks, where the language and context of incoming messages may vary widely. Furthermore, ML algorithms offer scalability, enabling NLP systems to handle diverse and evolving datasets with ease. For these reasons, employing ML algorithms is regarded as the optimal choice for the tasks in the model.

a. Naive Bayes: The Naive Bayes algorithm functions as a learning method derived from Bayes' theorem. Despite its seemingly straightforward design, it is capable of processing datasets with irregular structures. Bayes' Theorem is mathematically represented by the equation:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}. \quad (1)$$

The posterior probability of class A given characteristic B is denoted by P(A|B) in this instance. The probability of spotting feature B given class A is denoted by P(B|A). The evidence probability, or P(B), serves as a normalizing factor. P(A) is the prior probability of class A. Given the class label, the Naive Bayes classifier makes the assumption that the characteristics are conditionally independent. This makes calculating the likelihood easier:

$$P(B_1, B_2, \dots, B_n|A) = P(B_1|A) \cdot P(B_2|A) \cdot \dots \cdot P(B_n|A). \quad (2)$$

First, calculate the prior probability $P(A)$ for each class A in the training data. Then, calculate the likelihood $P(B|A)$ for each feature B given each class A. This can be done using different probability distributions such as Gaussian (as default in our model) or multinomial. Next, use Bayes' theorem to calculate the posterior probability $P(A|B)$ for each class A given the features B. This involves multiplying the prior probability with the likelihood and normalizing by the evidence probability:

$$P(A|B) = \frac{P(B|A) \times P(A)}{P(B)}. \quad (3)$$

Finally, assign the class label with the highest posterior probability as the predicted class for the given input features B:

$$\hat{A} = argmax_A P(A|B). \quad (4)$$

b. Passive Aggressive Classifier: The Passive Aggressive Classifier is an online learning algorithm used for binary classification tasks, particularly in scenarios where data streams continuously and there is a need to adapt to changes quickly. The primary concept behind the Passive Aggressive algorithm is that it makes aggressive updates when mistakes occur and passive updates when the prediction is correct. The algorithm begins by initializing a weight vector w and a bias term b to zero. During training, for each training example (x_i, y_i), where x_i is the input feature vector and y_i is the corresponding true label (either +1 or -1).

First, the algorithm computes the predicted label $\hat{y}_i$ using the current weight vector and bias term:

$$\hat{y}_i = sign(w \,.\, x_i + b). \quad (5)$$

Next, it calculates the loss incurred by the prediction using a loss function, typically the hinge loss:

$$l_i = \max(0, 1 - \, y_i \,.\, \hat{y}_i). \quad (6)$$

If the prediction is incorrect (i.e., $y_i \neq \hat{y}_i$), the algorithm updates the weight vector and bias term to correct the mistake. The update rule is based on the

aggressiveness of the update, determined by a hyperparameter C. The update is performed as follows:

$$w \leftarrow w + a.y_i.x_i, \tag{7}$$

where α is the learning rate, determined based on the loss and the aggressiveness parameter C. Similarly, the bias term is updated:

$$b \leftarrow b + a.y_i. \tag{8}$$

c. **Decision Tree Classifier:** Decision Tree Classifier is one another popular ML algorithm used for both classification and regression tasks. Mathematically, the decision tree algorithm involves calculating impurity measures and recursively partitioning the feature space based on these measures. The splitting criterion determines the feature and value that maximize the reduction in impurity at each node. Let's denote D as the dataset at a particular node, X_i as the i-th feature, v as the split value, and C as the set of possible class labels.

Based on the values of the input features, the decision tree method divides the feature space into smaller sections recursively. The decision to divide the dataset into two or more subsets is determined at each node of the tree based on the value of a characteristic. This procedure continues until a stopping criterion such as reaching a purity level, a maximum tree depth, or a minimum number of samples per leaf node is met. To choose the best feature and value for splitting the dataset at each node, a splitting criterion such as entropy or Gini impurity is used. The Gini Impurity measure calculates the likelihood that an element selected at random would be wrongly classified if its labels were randomly assigned based on the subset's label distribution. The

Gini impurity $I_G(D)$ for a dataset D is calculated as where $p(c)$ is the probability of class c in dataset D.

Entropy measures the level of impurity or uncertainty in a dataset. It is calculated as the sum of the probabilities of each class multiplied by the logarithm of the probability as in the below:

$$I_G(D) = 1 - \sum_{c \in C} p(c)^2. \tag{9}$$

The information gain IG measures the reduction by splitting the dataset D using a particular feature X_i and split value v. It is calculated as where k is the number of partitions resulting from the split, D_j is the subset of D that corresponds to the j-th partition, and $|D_j|$ denotes the number of instances in the subset D_j:

$$IG(D, X_i, v) = I_G(D) - \sum_{j=1}^{k} \frac{|D_j|}{|D|} I_G(D_j). \tag{10}$$

The decision tree algorithm selects the feature X_i and splits value v to maximize the information gain at each node. This process is repeated recursively for each subset until the stopping criteria are met. Once the tree is constructed, decision rules are formed based on the path from the root node to each leaf node. These decision rules define the conditions under which instances are classified into different classes. To classify a new instance, it is routed down the decision tree from the root node to a leaf node according to its feature values. The class label assigned to the leaf node that the instance reaches becomes its predicted class.

d. Random Forest Classifier: RF is a classification algorithm that tries to perform the classification and regression processes in the best possible way by using the advantages of multiple decision trees to create models that are more compatible and produce better results.

The importance of a feature f_i (denoted as fi_i) is calculated as the sum of the importance of all nodes (ni_j) where feature i is used for splitting, divided by the sum of importance values of all nodes in the tree:

$$fi_i = \frac{\sum_{j:i\ feature\ j\ node} ni_j}{\sum_{k\ \in\ all\ nodes} ni_k}. \tag{11}$$

Similarly, the importance of a node j (denoted as ni_j) is calculated based on its contribution to the overall decision tree. After calculating the importance of each feature, the values are normalized to ensure they range between 0 and 1. This normalization helps in comparing the relative importance of different features:

$$norm\ fi_i = \frac{fi_i}{\sum_{j\ \varepsilon\ allfeatures} fi_j}. \tag{13}$$

The final feature importance at the Random Forest level is obtained by averaging the normalized feature importance values from all trees in the ensemble:

$$RFfi_i = \frac{\sum_{j\ \in\ all\ trees} norm\ fi_{ij}}{\mathrm{T}}, \tag{13}$$

where $RFfi_i$ is the feature importance calculated from all trees in the Random Forest model, $norm\ fi_{ij}$ is the normalized feature importance for feature i in tree j, and T is the total number of trees in the Random Forest.

RF algorithm can also be evaluated for classifying categorical data. The algorithm mitigates overfitting problems with techniques such as bootstrapping and feature randomness. Bootstrapping involves creating multiple random samples with replacements from the original dataset, while Feature Randomness randomly selects a subset of features at each node split, reducing correlation between trees and improving generalization.

e. Support Vector Machine: Using kernel functions to identify the ideal hyperplane that optimizes the margin between classes in a higher-dimensional space, SVM is a flexible approach that can handle both linearly and non-linearly separable data. Another definition of SVM is the capacity to handle complex data.

Finding the ideal hyperplane to maximally segregate data points of distinct classes is the fundamental idea behind support vector machines (SVM). SVM maximizes the margin of the distance between the hyperplane and the closest data points from each class (referred to as support vectors in order to improve the model's capacity for generalization. The equation represents the hyperplane in a binary classification problem with two classes (positive and negative):

$$w^T x + b = 0, \tag{14}$$

where w is the weight vector (normal to the hyperplane) and b is the bias term. For linearly separable data, as in our model, SVM aims to find the optimal hyperplane that maximizes the margin while ensuring that all data points are correctly classified. Mathematically, this can be formulated as an optimization

problem. SVM solves the constrained optimization problem minimizing the objective function:

$$\frac{1}{2}\|w\|^2, \tag{15}$$

subject to the constraints:

$$y_i(w^T x_i + b) \geq 1 \text{ for i = 1,2,...,n}, \tag{16}$$

where (x_i, y_i) are training samples with x_i as the input features and y_i as the corresponding class labels (-1 or 1).

By employing a kernel function to transfer the input characteristics into a higher-dimensional space, SVM can also handle non-linearly separable data. Sigmoid, polynomial, linear, and radial basis function (RBF) are examples of common kernel functions. SVM introduces slack variables ε_i to allow for occasional misclassifications in situations where the data is not perfectly separable. As a result, the soft margin SVM is developed, which strikes a compromise between maximizing the margin and lowering the classification error. SVM can implicitly translate the input features into a higher-dimensional space, where the data may become linearly separable, thanks to the kernel trick. In the original feature space, a non-linear decision boundary correlates to the decision border in this higher-dimensional space. The optimization problem of SVM can be reformulated into its dual form, which leads to the computation of Lagrange multipliers a_i. The decision function for classifying new data points involves computing the dot product between the weight vector w and the feature vector of the new data point in the feature space. Once the optimal

hyperplane is found during training, SVM predicts the class of new data points based on which side of the hyperplane they fall on.

f. K-Nearest Neighbor: The k-NN algorithm calculates the distance of the new data to be added to the existing data sets and performs the classification process by looking at the K number of nearest neighbors. For distance calculations, it uses Euclidean, Minkowski, and Manhattan distance calculations. It is a popular classification algorithm that is still used today because its algorithm structure is resistant to old, noisy training data.

Classifying a new data point according to the majority class of its k nearest neighbors in the feature space is the fundamental notion behind k-NN. In other words, the class of the new data point is determined by the class labels of its nearest neighbors. The choice of distance metric plays a crucial role in k-NN. Common distance metrics include Euclidean distance, Manhattan distance, and Minkowski distance. Euclidean distance is the most commonly used (also in our model) distance metric and is calculated as:

$$\sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}, \tag{17}$$

where x_i and y_i are the i - th features of two data points. In the training phase of k-NN, the algorithm simply stores all the training data points along with their corresponding class labels. In the prediction phase, when a new data point is given, the algorithm calculates the distance between the new data point and all the training data points using the chosen distance metric. The k nearest neighbors of the new data point are selected based on their distances. These

neighbors are the data points with the smallest distances to the new data point. Once the k nearest neighbors are identified, the algorithm counts the number of data points in each class among these neighbors. The class with the highest count becomes the predicted class for the new data point. In the case of ties (i.e., when two or more classes have the same number of neighbors), k-NN uses a predefined decision rule to break the tie. Common decision rules include selecting the class with the smallest distance-weighted sum or selecting the class based on a predetermined order. The parameter k, which represents the number of nearest neighbors to consider, is a hyperparameter that needs to be selected prior to training. The choice of k can significantly impact the performance of the k-NN algorithm and is often determined using cross-validation techniques. Once the majority class is determined based on the class labels of the k nearest neighbors, k-NN predicts the class of the new data point as the majority class.

g. **Logistic Regression:** Logistic Regression is a statistical method used for binary classification tasks, aiming to predict the likelihood that an instance belongs to a specific class. Contrary to its name, logistic regression is a classification algorithm rather than a regression one. It is particularly useful for handling linearly separable data or scenarios with a high number of features.

Logistic regression models the relationship between the independent variables (features) and the probability of the dependent variable (target class) using the sigmoid function, also known as the logistic function. The sigmoid function is defined as:

$$\sigma(z) = \frac{1}{1 + e^{-z}}, \tag{18}$$

where z is a linear combination of the features and model coefficients. The coefficients (weights), $w_0, w_1, w_{2,} \dots, w_n$ are associated with each feature, and $x_0, x_1, x_{2,} \dots, x_n$ are the feature values in the equation below:

$$\text{z} = w_0 + w_1 x_1 + w_2 x_2 + \cdots + w_n x_n. \tag{19}$$

The logistic regression model is trained using a method called Maximum Likelihood Estimation (MLE) or optimization techniques like gradient descent. The goal is to find the optimal values for the coefficients $w_0, w_1, w_{2,} \dots, w_n$ that minimize the error between the predicted probabilities and the actual class labels in the training data. Once the model is trained, it can predict the probability that a new instance belongs to the positive class (class 1) using the sigmoid function. By choosing a threshold (typically 0.5), the predicted probability can be converted into a binary prediction: if the predicted probability is greater than the threshold, the instance is classified as belonging to the positive class; otherwise, it is classified as belonging to the negative class. Logistic regression models can be regularized to prevent overfitting by adding a regularization term to the cost function during training. The two most common types of regularization used are L1 regularization (Lasso) and L2 regularization (Ridge, as default in our model), which penalize large coefficients and encourage simpler models.

h. Bagging Ensemble Learning: Bagging, or Bootstrap Aggregating, aims to improve the stability and accuracy of machine learning models by combining

the predictions of multiple base learners trained on different subsets of the training data.

The process begins with initializing a set number of base learners. In the model, 11 different algorithms are used. Each base learner is trained on a bootstrap sample of the training data, where instances are randomly sampled with replacement. This allows the bootstrap sample to have the same size as the original training set and potentially contain duplicate instances. Each base learner is trained independently using a specified learning algorithm, in the model Decision Tree, to learn the mapping from features to target labels. Since base learners are trained on different subsets of data, they can be trained in parallel without any dependency on each other. Once all base learners are trained, predictions are made for each individual learner. For regression tasks, the final prediction is often determined by averaging the predictions from all base learners. For classification tasks, predictions can be combined through various methods, including majority voting, soft voting (where each base learner provides a probability distribution over the classes), or weighted voting (where base learners' predictions are weighted based on their performance). The bagging ensemble outputs the final prediction for each instance based on the chosen aggregation method, typically through majority voting. This technique effectively reduces overfitting and improves the generalization of machine learning models, making it a popular choice in ensemble learning.

i. **Hard Voting Ensemble Learning:** In Hard Voting, multiple base models are trained independently on the training data, and the final prediction is

determined by a majority vote among the predictions of the base models. Here are the mathematical steps for hard voting:

1. *Train Base Models:* Train multiple base models, denoted as M, using the training dataset. These base models can be any ML algorithms, such as decision trees, SVM, or neural networks. Each base model m_i learns to predict the target variable based on the input features. The 11 algorithms are used in the model, excluding Hard Voting itself.
2. *Generate Predictions:* After training the base models, use each base model to generate predictions for the test dataset. For each instance in the test dataset, each base model m_i will output a prediction $\hat{y}_i$.
3. *Voting:* Combine the predictions of all base models using a majority voting scheme. The final prediction for each instance is determined by selecting the class that receives the most votes among the predictions of the base models.

Mathematically, let $\hat{y}_{ij}$ denote the prediction of the jth base model for the i-th instance in the test dataset, where i = 1, 2, …, N and j = 1, 2, …, M. Each prediction $\hat{y}_{ij}$ belongs to a set of possible classes $C = \{C_1, C_2, \ldots, C_K\}$. Then, the final prediction $\hat{y}_i$ for the i th instance can be obtained by taking the majority vote:

$$\hat{y}_i = argmax_{c \in C} \sum_{j=1}^{M} 1(\hat{y}_{ij} = c), \tag{20}$$

where 1(.) is the indicator function that returns 1 if the condition is true and 0 otherwise, the argmax function returns the class with the highest sum of votes among the predictions of the base models for the i-th instance.

j. **Stacking Ensemble Learning:** Stacking, also known as stacked generalization, combines multiple base models using a meta-learner to make final predictions.

In the steps for stacking, first, the training data is divided into k folds for cross-validation. Next, M diverse base models are trained on different k-1 folds of the training data. Each base model learns to predict the target variable based on the input features. In this model, all algorithms are included as base models, except for Stacking itself. For each base model, the remaining fold (hold-out fold) is used to generate predictions for the instances in that fold. These predictions serve as the input features for the meta-learner. The predictions of all base models are then combined into a matrix X, where each row corresponds to an instance in the hold-out fold, and each column corresponds to a base model's prediction. A meta-learner or blender model is then trained on the combined predictions X and the true labels of the instances in the hold-out fold. The meta-learner learns to predict the target variable based on the combined predictions of the base models. The StackingClassifier algorithm is used for this purpose. These steps are repeated for each fold of the training data, so that each instance is used as part of the hold-out fold once. Finally, after training the meta-learner on all k hold-out folds, the predictions of the base models on the test data are used to generate predictions for each base model. These predictions are then combined using the trained meta-learner to obtain the final predictions for the test data.

k. **AdaBoosting Ensemble Learning:** Adaptive Boosting, or AdaBoost, is a well-liked ensemble learning method for regression and classification

applications. It functions by fusing together several weak learners, usually decision trees, to produce a powerful learner. In the model, AdaBoost iteratively trains weak learners using a weighted distribution of training instances. It adjusts the distribution based on the performance of each weak learner and combines their predictions to form a strong classifier. The final classifier gives higher weights to the predictions of weak learners that perform better on the training data. This process allows AdaBoost to focus on the training instances that are difficult to classify.

The process begins with initializing the distribution D_1(i) = 1/n for all training instances (i = 1,2,...,n), where n is the total number of training instances. During iterative training, for t = 1,2,...,T, at each iteration t, the current distribution (D_t) is provided to a weak learner, which then produces a weak hypothesis (h_t: X→[-1,1]), where X is the input space. The distribution is updated based on the performance of the weak learner. First, the weighted sum of the errors is calculated:

$$r_t = \sum_{i=1}^{n} D_t(i) y^{(i)} h_t\left(x^{(i)}\right) \in [-1,1], \tag{21}$$

where ($y^{(i)}$) is the true label of the i-th training instance. The strength of the weak learner's update is then determined by a function:

$$\frac{1}{2} \ln \frac{1+r_t}{1-r_t} \in \mathrm{R}, \tag{22}$$

where ln denotes the natural logarithm. The distribution for the next iteration is updated as in Equation 22.

Finally, the weak learners' predictions are combined to form the final classifier with the function:

$$D_{t+1}(i) \propto D_t(i) \exp\left(-a_t y^{(i)} h_t\left(x^{(i)}\right)\right), \quad (23)$$

where H(x) is the final prediction for input x. In H(x) function, $sign$ is the sign function, and T is the total number of iterations as in the below:

$$\mathrm{H(x)} = sign\ \left(\sum_{t=1}^{T} a_t h_t(x)\right). \quad (24)$$

l. Blending Ensemble Learning: Blending, also known as model averaging or stacking, is a specific type of ensemble learning technique where the predictions of multiple base models are combined using a meta-learner. Blending can handle complex relationships in the data and mitigate the weaknesses of individual models. Additionally, blending allows for flexibility in model selection. However, blending can be computationally expensive and requires careful tuning of the base models and meta-learner to optimize performance.

Initially, multiple base models are trained using different algorithms or variations of the same algorithm on the training data. These base models can be diverse, utilizing different algorithms, or they can be similar models trained on different subsets of the data or with different hyperparameters. In the model, 11 algorithms are employed, excluding Blending. Once the base models are trained, they are used to make predictions on the data, with each base model producing its own set of predictions for each instance in the dataset. In

blending, a meta-learner is trained to combine these predictions into a single prediction. The meta-learner takes the predictions of the base models as input features and learns to predict the final output. The meta-learner can be a simple model like logistic regression or a more complex model like a neural network. In this case, the model utilizes the XGBClassifier algorithm. The predictions made by the base models serve as features for the meta-learner, which is trained on these features along with the true labels of the training data. During training, the meta-learner learns to weigh the predictions of the base models in such a way that it maximizes the overall predictive performance. Once the meta-learner is trained, it can be used to make predictions on new, unseen data. The predictions of the base models are fed into the meta-learner, which then produces the final prediction.

3.3.3. RPA. The RPA system implemented for Turk Telekom's email management operates seamlessly within the company's infrastructure, specifically with the Siebel email system. Upon retrieving email data from Siebel, the RPA system efficiently processes and organizes the information for further analysis. Subsequently, the data is seamlessly transferred to the NLP system in the model, where it undergoes classification into distinct categories, primarily distinguishing between two broad classifications in Figure 3.: Complaint and Request.

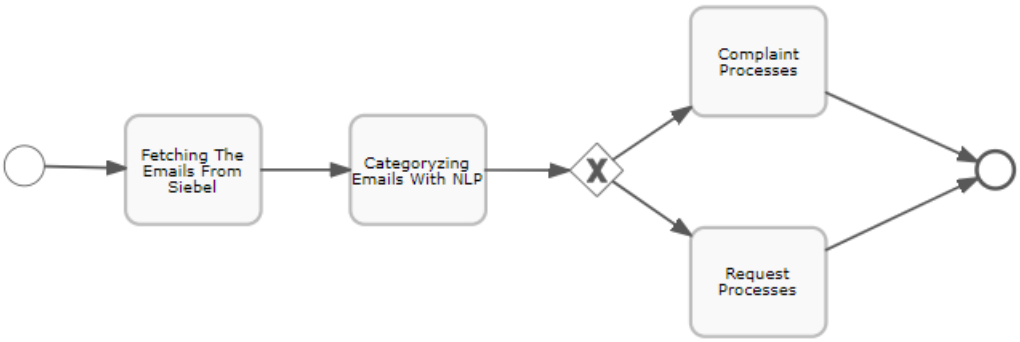

Figure 3. Initial processes of RPA and first categorization into two categories.

Following this initial categorization, the NLP system further refines the classification by assigning emails to one of nine specific subcategories: Kampanya/Paket, Ürün/Servis, Abonelik, Fatura ve Yükleme, Taahhüt, Fatura İtirazı, Tarife, CC Hat Satışı, and Ortak Marka Lansman. Each of these categories represents a specific type of inquiry or issue raised by customers.

Once the emails are classified into these detailed categories, the subsequent actions are determined based on the nature of the classification. For emails categorized as Requests, in Figure 4. the RPA system initiates automated processes tailored to address the specific request. These automated processes streamline various tasks related to customer inquiries or service requests, ensuring swift and efficient resolution.

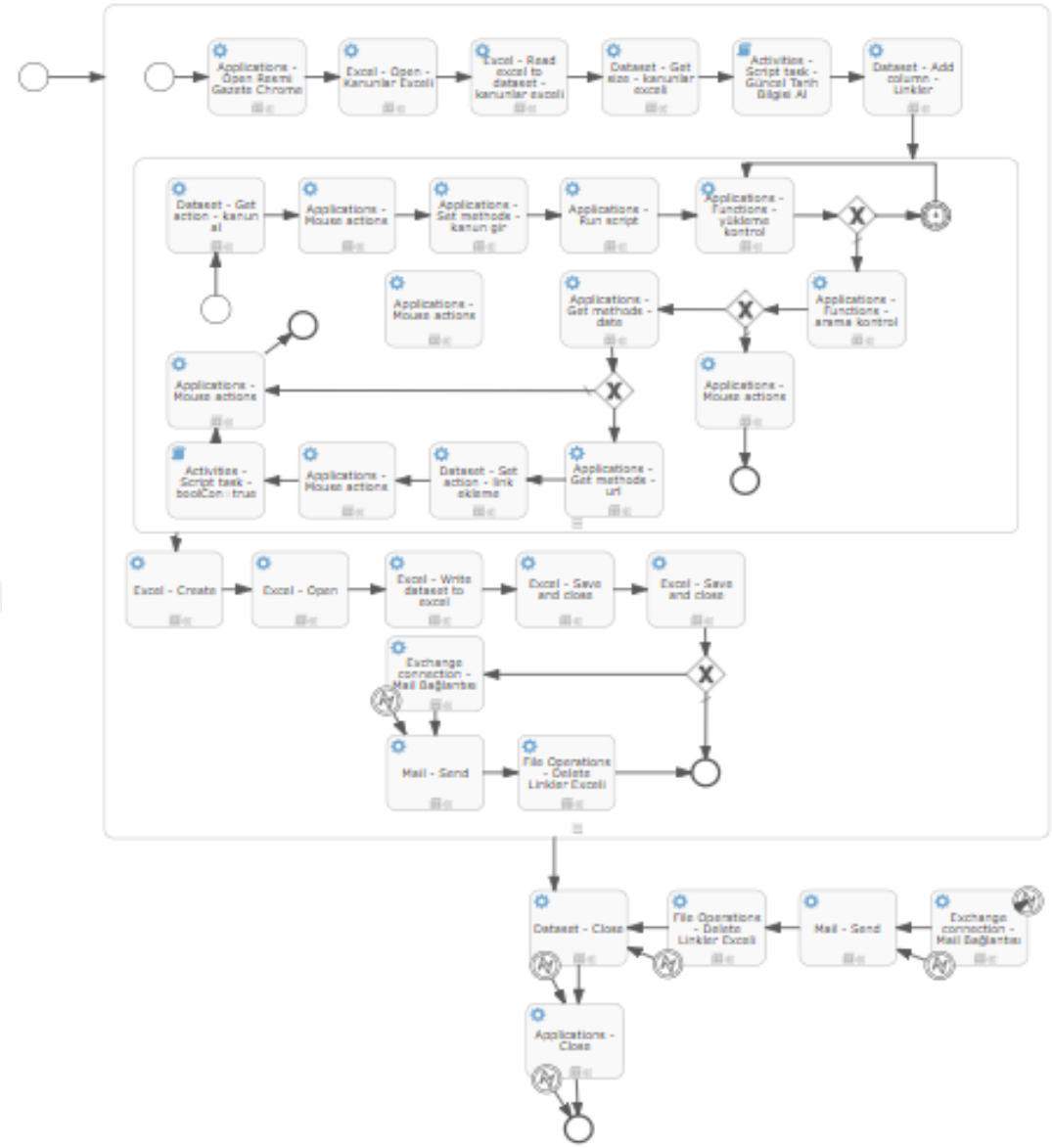

Figure 4. A sample process executed in one of nine Request sub-categories.

On the other hand, emails categorized as Complaints are directed to the Customer Services department for further handling and resolution. By automatically directing Complaint emails to the appropriate department, in Figure 5. the RPA system ensures that customer concerns are promptly addressed by the designated personnel, facilitating timely responses and effective resolution of issues.

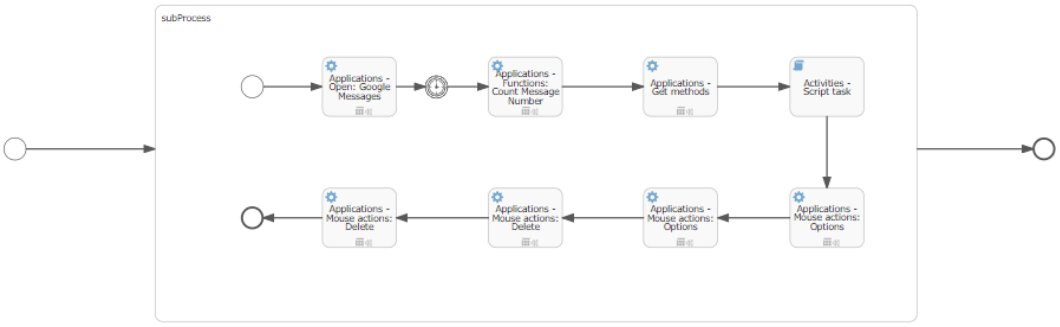

Figure 5. RPA flow of routing the Complaint emails to the Customer Services.

The RPA system not only streamlines email management procedures, but also manages all exceptions and human contacts inside the workflow. As the process manager, the RPA system autonomously monitors the entire process, applying advanced algorithms and decision-making logic to address any exceptions or human involvement requirements that may arise. When an exception arises, such as an email that falls outside of the specified classification categories or requires human intervention owing to its complexity, the RPA system automatically recognizes and manages the problem. To handle exceptions and human interactions, the RPA system follows predetermined rules and procedures to identify the best course of action. In circumstances that need human action, the system quickly directs the emails to authorized persons or departments for additional assessment. This ensures that all customer inquiries and complaints are promptly addressed by the appropriate personnel, maintaining high levels of customer satisfaction and service quality.

Before executing any automated processes, the RPA system rigorously verifies the accuracy and relevance of the classified data. Through comprehensive validation checks, the system ensures only accurate information is used in subsequent actions. This minimizes errors or inconsistencies that could affect customer service quality.

CHAPTER 4

Results

In our email classification model, a total of 12 ML algorithms were implemented to choose a superior algorithm to classify incoming emails better. For feature extraction, both Count Vectorizer and TF-IDF Vectorizer methods were employed. Through comparative analysis, it was determined that the TF-IDF Vectorizer yielded superior results. Building upon this insight, the efficacy of TF-IDF Bigram and TF-IDF Trigram Vectorizers was further explored. Each algorithm and extraction method was executed five times to ensure robustness, with average outcomes calculated.

The results, including ROC AUC score, sensitivity, and specificity values, along with accuracy scores, are presented in Table 1, providing a clear overview of the effectiveness of the different algorithms and extraction techniques employed. This comprehensive analysis highlights the comparative performance and reliability of each approach, ensuring that the best possible model is chosen for classifying incoming emails.

Algorithms	Accuracies						
	CountVectorizer	TFIDF Vectorizer	TFIDF Bigram	TFIDF Trigram	ROC AUC Score	Specificity	Sensitivity
Naive Bayes	0.971	0.921	0.923	0.965	0.939	0.994	0.883
Passive Aggressive Classifier	0.973	0.983	0.976	0.989	0.985	0.992	0.978
Decision Tree Classifier	0.948	0.753	0.781	0.972	0.965	0.981	0.948
Random Forest Classifier	0.969	0.752	0.774	0.972	0.965	0.981	0.948
SVM	0.968	0.981	0.986	0.990	0.988	0.993	0.982
K-NN	0.819	0.824	0.953	0.867	0.967	0.923	0.951
Logistic Regression	0.976	0.986	0.981	0.982	0.977	0.989	0.966
Ensemble: Bagging	0.959	0.955	0.976	0.990	0.988	0.993	0.982
Ensemble: Hard Voting	0.983	0.963	0.978	0.991	0.987	0.994	0.980
Ensemble: Stacking	0.979	0.989	0.975	0.991	0.989	0.993	0.985
Ensemble: AdaBoosting	0.952	0.926	0.971	0.976	0.985	0.996	0.976
Ensemble: Blending	0.985	0.969	0.978	0.990	0.988	0.992	0.984

Table 1. Comparison of the ML algorithms' accuracies.

Figure 6. Total success and failed amounts of automation execution.

The email classification automation operates on a 30-minute interval throughout the day, ensuring continuous processing of incoming emails. Monitoring the performance, daily reports, which are generated in the management panel of the RPA platform called Orchestrator, are reviewed and detailed insights are analyzed using the Kibana platform.

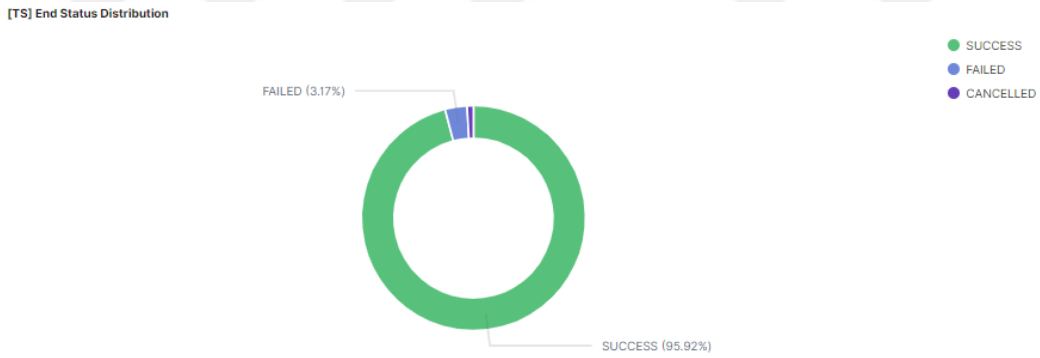

Figure 7. Success and failure proportion of the automation.

Leveraging the data, two comprehensive tables are compiled: Figure 6. illustrating the respective quantities of these occurrences, and Figure 7. depicting the proportions of success and failure instances.

4.1. SWOT Analysis

The SWOT framework, which stands for Strengths, Weaknesses, Opportunities, and Threats, is commonly utilized to evaluate the fundamental aspects of businesses. This framework offers comprehensive insights into various aspects that need consideration. By meticulously analyzing each block in connection with our model, this 2x2 matrix provides a clear understanding of our current position and informs us about areas that require attention.

Table 2. SWOT Analysis

Strength	**Weakness**
• The Automated Email Classifying is done by creating an NLP-RPA Automation using Robusta. • The automation allows us to classify unstructured emails using AI Technologies such as NLP, ML, and RPA.	• Although predominantly automated, the process may occasionally necessitate human intervention. • Potential disruptions such as power outages or system shutdowns can impact the automation.
Opportunities	**Threats**
• It's estimated that upwards of 75% to 80% of organizational emails can be effectively processed through this automation. • Given its critical role, email classification stands as a fundamental necessity for most organizations. • The company has the opportunity to expand its global operations.	• However, there exists a risk of misclassifying emails into incorrect categories. • The success of this initiative can attract numerous competitor companies and imitative brands, posing potential threats to market dominance.

Here's an overview of the SWOT analysis performed, depicted in Table 2:

1. *Strengths:* Utilizing Robusta, our NLP-RPA Automation facilitates Automated Email Classification, employing advanced AI Technologies like NLP, ML, and RPA to categorize unstructured emails.

2. *Weaknesses:* Even after being automated, human help might be needed but that is with every robot. Also, technical dependencies could become an issue, if ever came up.
3. *Opportunities:* 80% of the emails in the organization can be processed. Allows the company to expand into global areas.
4. *Threats:* There will always be a chance of misclassifying emails. Because of the success, many companies are trying to create the same now so competitors arrive.

CHAPTER 5

Conclusion

In conclusion, the integration of NLP and RPA in call center operations marks a significant advancement. The development of an email parsing automation system has demonstrated the effectiveness of this approach in streamlining workflows and enhancing efficiency. The successful deployment of RPA tools like Robusta further reinforces this synergy, automating tasks tailored to the call center environment. Real-world use cases and benchmarks highlight tangible benefits, from improved productivity to enhanced customer satisfaction. Looking ahead, the convergence of NLP and RPA will continue to reshape call center dynamics, offering opportunities for continuous improvement and unparalleled customer experiences. Integrating NLP and RPA isn't just a technological advancement but a strategic imperative for all sectors in the digital age, driving efficiency, effectiveness, and excellence.

REFERENCES

Aery M. and Chakravarthy S., 2005. "eMailSift: eMail classification based on structure and content," *Fifth IEEE International Conference on Data Mining (ICDM'05)*, Houston, TX, USA, pp. 8 pp.-, https://doi: 10.1109/ICDM.2005.58.

Akyıldız M., 2007. Yazılım Projeleri Ölçüm Sonuçları Veri tabanıb Oluşturulmasu ve Yeni Yazılım Projelerinin Maliyet Tahmininde Kullanımı. Yıldız Teknik Üniversitesi Fen Bilimleri Enstitüsü.

Albayrak, A., 2020. Doğal Dil İşleme Teknikleri Kullanılarak Disiplinler Arası Lisansüstü Ders İçeriği Hazırlanması. Bilişim Teknolojileri Dergisi, 13(4), 373-383. https://doi.org/10.17671/gazibtd.714447

Arevian, G., 2007. Recurrent neural networks for robust real-world text classification. Proceedings of the International Conference on Web Intelligence, Nov. 2-5, IEEE Xplore Press, Fremont, CA, pp: 326-329. DOI: 10.1109/WI.2007.126

Aydin G., Hallaç İ.R., Türkçe Metinlerde Otomatik Konu Tespiti, Fırat Üniversitesi Mühendislik Bilimleri Dergisi, 33 (2), 599–606, 2021.

Ayodele, T., R. Khusainov and D. Ndzi, 2007. “Email classification and summarization: A machine learning approach”. Proceedings of the IET Conference on, Wireless, Mobile and Sensor Networks, Dec. 12-14, IEEE Xplore Press, Shanghai, China, pp: 805-808.

Beseiso, M., A.R. Ahmad and R. Ismail, 2012. A new architecture for email knowledge extraction. Int. J. Web Semantic Technol., 3: 1-10. DOI: 10.5121/ijwest.2012.3301

Borandağ E. ,Yücalar F. ,Şahinaslan Ö. 2013. Yazılım Projelerinde Büyüklük Tahmini. Maltepe Üniversitesi, Mühendislik ve Doğa Bilimleri Fakültesi, Yazılım Mühendisliği Bölümü, Maltepe Üniversitesi, Bilişim Bölümü.

Braidotti, R., 1997. Meta(l)morphoses. *Theory, Culture & Society, 14*(2), 67-80. https://doi.org/10.1177/026327697014002007

C. Bataller , A. Jacquot, S.R. Torres, 2019. "EP3215900B1 Raporu", https://patents.google.com/patent/EP3215900B1/de?oq=Sergi+o+Ra%C3%BAl+TORRES.

C. Bataller, A. Jacquot, 2017. "US9555544B2 Raporu ", https://patents.google.com/patent/US9555544?oq=robotik+process+automation.

Çağataylı M., Çelebi E., 2015. The effect of stemming and stop-word-removal on automatic text classification in Turkish language. Arik S, Huang T, Lai WK, Liu Q, editors. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), 9489, 168–76.

Çalışkan, L. S., & Kıran, S., 2020. İş Süreçlerinin Otomasyonunda RPA'nın Faydaları. Yönetim Bilişim Sistemleri Dergisi, 6(1), 1-13.

Carmona-Cejudo, J.M., G. Castillo, M. Baena-Garcia and R. Morales-Bueno, 2011. "A comparative study on feature selection and adaptive strategies for email foldering". Proceedings of the 11th International Conference on Intelligent Systems Design and Applications, Nov. 22-24, IEEE Xplore Press, Cordoba, pp: 1294-1299. DOI: 10.1109/ISDA.2011.6121838

Chan, J., I. Koprinska and J. Poon, 2004. Co-training with a single natural feature set applied to email classification. Proceedings of the International Conference on

Web Intelligence, Sept. 20-24, IEEE Xplore Press, pp: 586-589. DOI: 10.1109/WI.2004.10135

Damar H., 2022. Robotik Süreç Otomasyonu (RPA) Nedir?. https://medium.com/digigeek/robotik-s%C3%BCre%C3%A7-otomasyonu-rpa-nedir-a20fb65ac1f1/ 05.03.2021.

Deniz A., Kiziloz H.E., 2017. Effects of various preprocessing techniques to Turkish text categorization using n-gram features. 2nd International Conference on Computer Science and Engineering, UBMK 2017, 655– 60.

Dharmadhikari S.C., Ingle M., Kulkarni P., Empirical Studies on Machine Learning Based Text Classification Algorithms. Advanced Computing, 2 (6), 161, 2011.

Dredze, M.H., 2009. Intelligent Email: Aiding Users with Ai. 1st Edn., University of Pennsylvania, ISBN-10: 1109227795, pp: 227.

Duan, G., Shen, Z. and Liu, R., 2019. An MBD based framework for relative position accuracy measurement in digital assembly of large-scale component, *Assembly Automation*, Vol. 39 No. 4, pp. 685-695. https://doi.org/10.1108/AA-04-2018-062

Erdoğan S., 2020. Robotik Süreç Nasıl Yapılır?. https://keytorc.com/blog/rpa-uygulamalari-nasilyapilir/ 04.03.2024.

Erşahin B., Aktaş Ö., Kilinç D., Erşahin M., A hybrid sentiment analysis method for Turkish. Turkish Journal of Electrical Engineering & Computer Sciences, 27, 1780–93, 2019.

Ferrando, F., 2014. Is the post-human a post-woman? Cyborgs, robots, artificial intelligence and the futures of gender: a case study. *Eur J Futures Res* **2**, 43. https://doi.org/10.1007/s40309-014-0043-8

Figueiredo, A.S. and Pinto, L.H., 2021. Robotizing shared service centres: key challenges and outcomes, *Journal of Service Theory and Practice*, Vol. 31 No. 1, pp. 157-178. https://doi.org/10.1108/JSTP-06-2020-0126

Fong, S., D. Roussinov and D.B. Skillicorn, 2008. "Detecting word substitutions in text". Knowledge Data Eng. IEEE Trans., 20: 1067-1076. DOI: 10.1109/TKDE.2008.94

Ganesh et al., 2018. "US2018197123 Raporu", https://patents.google.com/patent/US20180197123A1/en?oq=US2018197123 .

Golizadeh, H., Hosseini, M.R., Martek, I., Edwards, D., Gheisari, M., Banihashemi, S. and Zhang, J., 2020. Scientometric analysis of research on "remotely piloted aircraft": A research agenda for the construction industry, *Engineering, Construction and Architectural Management*, Vol. 27 No. 3, pp. 634-657. https://doi.org/10.1108/ECAM-02-2019-0103

Gür, Y. E., Ayden, C., & Yücel, A., 2019. YAPAY ZEKÂ ALANINDAKİ GELİŞMELERİN İNSAN KAYNAKLARI YÖNETİMİNE ETKİSİ. Fırat Üniversitesi Uluslararası İktisadi Ve İdari Bilimler Dergisi, 3(2), 137-158.

Gurcan F., Multi-Class Classification of Turkish Texts with Machine Learning Algorithms, ISMSIT 2018 - 2nd International Symposium on Multidisciplinary Studies and Innovative Technologies, Proceedings, 1– 5, 2018.

Haiyi, Z. and D. Li, 2007. Naïve bayes text classifier. Proceedings of the International Conference on Granular Computing, Nov. 2-4, pp: 708-708.

Hirschberg J. and Manning C. D., 2015. "Advances in natural language processing". Science, vol. 349, no. 6245, pp. 261–266. https://doi:10.1126/science.aaa8685

Huettinger, M. and Boyd, J.A., 2020. "Taxation of robots – what would have been the view of Smith and Marx on it?", *International Journal of Social Economics*, Vol. 47 No. 1, pp. 41-53. https://doi.org/10.1108/IJSE-11-2018-0603

Kadhim A.I., Survey on supervised machine learning techniques for automatic text classification, Artificial Intelligence Review, 52 (1), 273–92, 2019.

Kandimalla B., Rohatgi S., Wu J., Giles C.L., Large Scale Subject Category Classification of Scholarly Papers With Deep Attentive Neural Networks, Frontiers in Research Metrics and Analytics. 5, 600382, 2021.

Khan A., Baharudin B., Lee L., Khan K., A review of machine learning algorithms for text-documents classification. Journal of Advances in Information technology, 1 (1), 4–20, 2010.

Kilimci Z.H., Akyokus S., Deep learning- and word embedding-based heterogeneous classifier ensembles for text classification, Complexity, 2018.

Kilimci Z.H., Akyokus S., The Evaluation of Word Embedding Models and Deep Learning Algorithms for Turkish Text Classification, UBMK 2019 - Proceedings, 4th International Conference on Computer Science and Engineering, 548–53, 2019.

Küçük D., Arıcı N., A Literature Study on Deep Learning Applications in Natural Language Processing, International Journal of Management Information Systems and Computer Science, 2 (2),76–86, 2018.

Lyne, I., Ngin, C. and Santoyo-Rio, E., 2018. "Understanding social enterprise, social entrepreneurship and the social economy in rural Cambodia", *Journal of Enterprising Communities: People and Places in the Global Economy*, Vol. 12 No. 3, pp. 278-298. https://doi.org/10.1108/JEC-11-2016-0041

Maček, A., Murg, M. and Čič, Ž.V., 2020. How Robotic Process Automation is Revolutionizing the Banking Sector, Dirsehan, T. (Ed.) *Managing Customer Experiences in an Omnichannel World: Melody of Online and Offline Environments in the Customer Journey*, Emerald Publishing Limited, Leeds, pp. 271-286. https://doi.org/10.1108/978-1-80043-388-520201020

Madakam, S., Holmukhe, R.M., Jaiswal, D.K., 2019. "The future digital work force: robotic process automation (RPA)". J. Inf. Syst. Technol. Manag. 16. https://doi:10.4301/s1807-1775201916001

Noyan M., 2019. https://merveenoyan.medium.com/do%C4%9Fal-dil-i%CC%87%C5%9Fleme-natural-language-processing-2d7c72daf245 06.03.2024

Öztürkmenoğlu O., Alpkoçak A., 2012. Comparison of different lemmatization approaches for information retrieval on Turkish text collection. INISTA 2012 - International Symposium on INnovations in Intelligent SysTems and Applications, 1–5.

Pitts, F. H., & Dinerstein, A. C., 2017. Corbynism's conveyor belt of ideas: Postcapitalism and the politics of social reproduction. *Capital & Class*, 41(3), 423-434. https://doi.org/10.1177/0309816817734487

Priyadarshi, P. and Premchandran, R., 2022. "Insecurity and turnover as robots take charge: impact of neuroticism and change-related uncertainty", *Personnel Review*, Vol. 51 No. 1, pp. 21-39. https://doi.org/10.1108/PR-06-2019-0310

Provost, J., 1999. Naive-Bayes vs. Rule-Learning in Classification of Email. AI-TR-99-284, Univ. Of Texas at Austin.

Ravi et al., 2017. “US20170352041 Raporu” https://patents.google.com/patent/US20170352041A1/en?oq=US20170352041.

Romero, D., Stahre, J., Wuest, T., Noran, O., Bernus, P., Fast-Berglund, A. and Gorecky, D., 2016. Towards an operator 4.0 typology: a human-centric perspective on the fourth industrial revolution technologies” *in International conference on computers and industrial engineering (CIE46) proceedings*, pp. 1-11, ISSN 2164-8670, ISSN 2164-8689.

Rose, N. S., Aicardi, C., & Reinsborough, M. T., 2016. Foresight report on future computing and robotics: A Report from the HBP Foresight Lab . https://kclpure.kcl.ac.uk/admin/files/86508137/KCLForesightLab_2016_Future_computing_robotics.pdf

Ruiz R. C., Ramírez A. J., Cuaresma M. J. E., Enríquez J. G., 2022. “Hybridizing humans and robots: An RPA horizon envisaged from the trenches”, *Computers in Industry*, Volume 138, 103615, ISSN 0166-3615, https://doi.org/10.1016/j.compind.2022.103615.

S. Sel and D. Hanbay, "E-Mail Classification Using Natural Language Processing," *2019 27th Signal Processing and Communications Applications Conference (SIU)*, Sivas, Turkey, 2019, pp. 1-4, doi: 10.1109/SIU.2019.8806593.

Salur M., Aydın İ., Jamous M., An ensemble approach for aspect term extraction in Turkish texts. Pamukkale University Journal of Engineering Sciences, 28 (5), 769-776, 2021.

Santos, F., Pereira, R. and Vasconcelos, J.B., 2020. Toward robotic process automation implementation: an end-to-end perspective, *Business Process*

Management Journal, Vol. 26 No. 2, pp. 405-420. https://doi.org/10.1108/BPMJ-12-2018-0380

Sharma S., Dixit S.D., Patel V., Kalmalı Z., Agrawal P., 2019. "US10354225B2 Raporu", https://patents.google.com/patent/US10354225B2/en?inventor=Sujata+Sharma&oq=Sujata+Sharma

Sotala, K., & Yampolskiy, R. V., 2013. Responses to catastrophic AGI risk: A survey (Technical Reports, 2013 (2)). *Berkeley, CA: Machine Intelligence Research Institute*.

Tahiroğlu B.T., Lemmatization and a lemmatization application for Turkish: elemanTR. RumeliDE Journal of Language and Literature Research, 24, 475–86, 2021.

Tham, K.-T., Ng, K.-W., & Haw, S.-C., 2023. "Phishing message detection based on keyword matching", *Journal of Telecommunications and the Digital Economy*, 11(3), 105–119. https://search.informit.org/doi/10.3316/informit.342481290943158

Uğurlu E., 2019. Fintech Robotik Süreç Otomasyonu Uygulamaları. https://fintechtime.com/2019/01/fintech-robotik-surec-otomasyonu-uygulamalari/ 05.03.2024.

Urbahs, A. and Zavtkevics, V., 2019. Oil spill remote monitoring by using remotely piloted aircraft, *Aircraft Engineering and Aerospace Technology*, Vol. 91 No. 4, pp. 648-653. https://doi.org/10.1108/AEAT-12-2017-0273

Wang, R., A.M. Youssef and A.K. Elhakeem, 2006. On some feature selection strategies for spam filter design. Proceedings of the Canadian Conference on

Electrical and Computer Engineering, (ECE' 06), IEEE Xplore Press, Ottawa, Ont., pp: 2186-2189. DOI: 10.1109/CCECE.2006.277770

Weijie, L., H. Chen, W. Cao and X. Zhou, 2012. An idea of setting weighting functions for feature selection. Proceedings of the 2nd International Conference on Cloud Computing and Intelligent Systems, (CIS' 12), IEEE Xplore Press, Hangzhou, pp: 690-695. DOI: 10.1109/CCIS.2012.6664263

Willcocks, Leslie, Lacity, Mary and Craig, Andrew, 2017. Robotic process automation: strategic transformation lever for global business services? Journal of Information Technology Teaching Cases . pp. 1-12. ISSN 2043-8869

Yang, W. and L. Kwok, 2012. Improving the automatic email responding system for computer manufacturers via machine learning. Proceedings of the International Conference on Information Management, Innovation Management and Industrial Engineering, Oct. 20-21, IEEE Xplore Press, Sanya, pp: 487-491. DOI: 10.1109/ICIII.2012.6340024

Yıldırım S., Yıldız T., A comparative analysis of text classification for Turkish language. Pamukkale University Journal of Engineering Sciences, 24 (5), 879–86, 2018.

Zhang, N. and Liu, B., 2019. "Alignment of business in robotic process automation", *International Journal of Crowd Science*, Vol. 3 No. 1, pp. 26-35. https://doi.org/10.1108/IJCS-09-2018-0018