Politecnico di Torino

Corso di Laurea
Master’s degree course in Engineering and Management
A.a. 2022/2023
Sessione di Laurea Dicembre 2023

Support ticket categorization through Latent Dirichlet Allocation

Supervisor:
Fabio Salassa

Candidate:
Dario Buongarzone

Abstract

In contemporary corporate environments, the efficient handling of service desk tickets is pivotal for ensuring smooth IT operations. Support tickets, encompassing a range of customer requests and issues, are essential communication tools between customers and support teams. Proper categorization and swift resolution of these tickets are crucial tasks, ensuring customer satisfaction, productivity, compliance with service level agreements, and cost efficiency. In this study, the potential of Latent Dirichlet Allocation (LDA), an unsupervised machine learning algorithm, was explored for automatic categorization of service desk queries. Leveraging firsthand experience as an IT intern at Lavazza and access to a substantial dataset of tickets, this research successfully implemented LDA using Dariah-DE Topics Explorer. The findings indicate that LDA holds promise in enhancing ticket resolution efficiency, potentially revolutionizing service desk operations. However, the system's effectiveness depends on the quality and diversity of the training dataset and requires continuous optimization. Future research could focus on integrating LDA within existing service desk software, augmenting their capabilities for streamlined and efficient ticket handling processes. This study lays the foundation for further advancements in automating service desk ticket categorization, aiming for more advanced and responsive service desk operations in the future.

Acknowledgments

I miei più sentiti ringraziamenti al mio relatore, il Prof. Fabio Salassa, per avermi ascoltato, avermi spronato a vedere il problema da varie prospettive e non chiudermi alla mia interpretazione, per la sua disponibilità e grande obbiettività durante l'ultimo sforzo della mia carriera universitaria.

Questo tassello della mia vita, questo capitolo finale di questo viaggio, lo devo in gran parte alle persone che l'hanno percorso al mio fianco.

Ai miei amici, per le notti in aula studio a studiare, per gli ultimi preziosi consigli i giorni prima degli esami, per la determinazione che mi istillavate senza neanche accorgevene. Grazie per aver camminato al mio fianco, grazie per tutti quei momenti di vita vera che mai dimenticherò.

Ad Allegra, la mia complice, grazie per avermi letto dentro, per aver notato parti di me di cui ignoravo l'esistenza, per averle coltivate e fatte crescere. Grazie per non aver mai smesso di crederci, non sarei così appagato da questo traguardo se non fosse stato condiviso insieme. È grazie a te se sono una persona migliore. Il tuo starmi vicino, vera, onesta e paziente, ha dato un nuovo senso a tutto.

Infine grazie ai miei genitori, per tutto il sostegno e la fiducia che avete riposto in me incondizionalmente, nelle mie scelte, nelle mie decisioni e modi di vivere, anche se mi portavano spesso troppo distante da voi. Grazie per l'esempio e la motivazione di mio padre, per la sua forza e la sua incrollabile determinazione, da sempre la mia ancora e punto fisso. Grazie all'umorismo, la gentilezza e l'amore di mia madre, senza di te non sarei l'uomo che sono oggi, il tuo cuore e la tua bontà sono il mio passato presente e futuro.

A tutte le persone con cui ho incrociato cammini, condiviso momenti e vissuto emozioni, agli amici presenti e non, ai parenti riuniti qui per me, un grazie di cuore a tutti voi.

Contents

Abstract 3
Acknowledgments 4
1 Introduction 7
2 Literature Review 9
2.1 Background 9
2.2 History of Natural Language Processing 11
2.3 Current and future use of topic modeling 13
2.4 Related work utilizing topic modeling algorithms 15
3 Theory 17
3.1 Preprocessing 19
3.1.1 Tokenization 20
3.1.2 Stemming or Lemmatization 21
3.1.3 Stop-words 21
3.1.4 Conversion and transformation into digits 22
3.2 Topic modeling 24
3.2.1 Latent Semantic Indexing 25
3.2.2 Latent Dirichlet Allocation 26
.... 29
4 Research objectives and research framework 32
5 Methodology 34
5.1 Data collection 34
5.2 Preparation of the dataset 35
5.3 Stop-word list 36
5.4 Parameters 37
6 Data results 38
6.1 Topics 38

6.2 Document-topic-distribution 40
6.3 Topic similarity 42
6.4 Document similarity 43
7 Data Discussion 44
8 Conclusion 47
9 References 49

1 Introduction

In our modern, fast paced, industry environment, efficient dealing of service desk tickets is essential for ensuring seamless IT operations.

The expression *support ticket* describes a request for aid from a customer, either internal or external to the firm, to a support team, usually external and bound to the corporation by a contract. These include customer complaints, incident reports, equipment related requests and software related problems and requests [1].

The perks of support tickets over other tools of communication such as e-mails, calls and chats are that all communications between users and the service desk team can be tracked, recorded and kept for years inside databases.

When a ticket is sent, its categorization and dispatching to IT experts are tasks of key importance: swift solving assures personnel wellbeing, improved productivity due to decreased idle time, conformity to service level agreements (SLAs) and better cost efficiency for the contract stipulated between the org and the service provider.
On the other hand, improper dispatching of requests may result in uneconomical reassignment and worse resource utilization, with adverse consequences graving mostly on the service desk team.

During the course of my experience in Lavazza, I've stumbled upon the problem of ticket resolution by external consultants. Drawing upon my exposure to corporate matters as an IT intern in this business environment and having access to the database of service desk requests. This dissertation will delve into the complications of employing a branch within text mining, such as topic modeling, to draw out and inspect the predominant topics and key words within a corpus of tickets.

Topic modeling, a machine learning method, is a widespread text corpus analysis technique that spots affinities between words in a collection of documents. The primary goal is to detect hypothetical themes and topics hidden in the files, giving users an overview of the core meaning hidden in the dataset. These topics are outputted in the form of an array of words, as illustrated in Figure 2, in the Theory chapter, with words arranged in decreasing order based on their chance of being found within the corpus.

Latent Dirichlet Allocation (LDA) was the method used in this experiment to uncover these hidden topics. LDA, rarely not commonly used as a synonym for topic modeling, is the chosen algorithm for this text mining research [2]. As we will further see in the methodology chapter, it makes uncovering the hidden general meaning of similar textual thanks to assigning to each document a string of words termed *topic.*

Each topic equals a distribution over the English vocabulary, derived from the collection of service desk tickets (the text corpus under analysis). The documents, on the other hand, are mix of these topics, automatically gathered by the model from the words contained inside each single topic and, originally, each document.

The entire research was made using *Dariah-de Topics Explorer*, an open-source software coded in python and encapsulating the LDA algorithm, but assisting the customer with a user-friendly experience, permitting the uploading a collection of plain text files and analyzing it by the means of the algorithm.

For the objective of this thesis, a database of corporate tickets was gathered. Afterwards, the documents underwent strict preprocessing before topics could be automatically generated by the script. Finally, the topics were output by the software and visioned, and a human verification and data analysis was employed.

2 Literature Review

2.1 Background

The increasing sheer amount of published journals, articles, and science papers each year carries with it a tremendous increase in fake and erroneous information. This matter poses a compelling challenge for researchers, even more so due to the overwhelming amount of unstructured data, such as digital images, videos, audios, and texts. Unstructured data is much more complex to use compared to structured data, which is more manageable and readable for machine learning software.

The current active global amount of internet users tops 4.5 billion, and this number is destined to rise. Given such a great online presence of users, sharing and using internet based data, it is inevitable that the gap between unstructured and structured data will further enlarge [3].This increasing divide entails a complicating factor in data analysis, as the inspection process gets more and more time-consuming and problematic for researchers.

Automated and unsupervised text mining offers a solution for handling this vast volume of unstructured data. These technologies facilitate better exploration and comprehension of such information, accentuating the importance of the question: to what degree can a machine learning algorithm disclose the hidden meaning of raw human text inside a service desk ticket to automate its categorization?

Studies show that approximately 80 percent of corporation-related information is unstructured, emphasising the urgency for such text mining techniques. Corporations can derive profits from these by automatically analysing their unstructured data and better utilising existing, but often overlooked, documents. For example, in consultancy firms, problems often arise where consultants can't reliably compare colleagues' previous discoveries and advice for similar cases. To respond efficiently to such challenges, various algorithms have been developed, with text mining often showing promising results.

It is beneficial to lay the foundation for a clear definition of topic modeling inside the aim of this dissertation, taking into consideration the numerous previously existing interpretations of the term. Topic modeling takes place as a sub-field within text mining, a concept described by Hotho et al. in their paper titled "A Brief Survey of Text Mining". Text mining, on one hand, can be split into three distinct sub-fields, each with its own explanation. The first sub-field, as defined by Hotho et al.[4], is information extraction. This sub-field is centered on extrapolating specific facts from textual data and represents a limited form of natural language understanding. In information extraction, researchers pre-establish the type of semantic information they are looking for in the text and afterwards extrapolate the meaningful portions while giving specific attributes. To better explain, consider an online price checker where the item and its previous prices are known. The task revolves around updating the price based on new available data, pursuing a series of intermediate steps such as sentence segmentation, tokenization, part-of-speech tagging, and identification of named entities such as organizational nouns. These steps are necessary not exclusively to information extraction, but also very much so for to topic modeling.

The second sub-field, as outlined by Hotho et al., is known as text data mining. In this sub-field, scripts and methods taken from machine learning and statistics are put in application to texts to discover patterns, similar to the goal in traditional data mining. Techniques like Information Extraction and Natural Language Processing (NLP) are deployed to extract meaningful information from documents. An explicative application of text data mining is topic modeling, where the objective is to extract distinct characteristics or themes from textual data. This method is conductive in categorizing and structuring large collections of text or extrapolating valuable information [7]. As elaborated in further paragraphs of the thesis, the key passages in topic modeling are the following: data collection; preprocessing; and the creation of topics, followed by the graphic and tabular display of results.

The third sub-field of text mining, named knowledge discovery in databases [4], is conducted as a systematic process mixing techniques from both previous steps, such as information extraction and text data mining. Its primary goal is to bring to light connections and hidden patterns within a dataset, utilizing methods from the first two definitions of text mining [4]. An explicative application of this method is in fraud detection, here, bank databases are continuously investigated. Even small and insignificant alterations in customers' spending patterns can warn of potential fraud. In these scenarios, raw data, including text files, first undergo pre-processing and modification deploying techniques from information extraction. Afterwards, the refined data are analyzed using text data mining algorithms. The outcome is then assessed and studied to identify potential changes in patterns. This comprehensive methodology permits detection of anomalies and uncovered trends inside data.

Between these three sub-fields of text mining, only the second sub-field holds meaningfulness for this thesis. However, some of the steps deployed in the first sub-field will be incorporated thanks to their widespread adoption when dealing with and employing topic modeling algorithms [4].

During his lecture at the University of Edinburgh, David Blei, a co-founder of Latent Dirichlet Allocation (LDA), was keen on emphasizing the essential role of topic modeling in organizing, visualizing, summarizing, searching, predicting, and understanding vast collections of documents undergoing machine learning analysis [5]. Blei emphasized that topic models have been proven useful plenty of times when interpreting various types of data, including images, social connections, and genetic information [5]. The primary goal of topic models is to discover the thematic structure from within the textual content. This also includes annotating documents and utilizing these annotations to visualize, organize, and summarize the results. The ultimate objective is to employ this process efficiently, even though often dealing with millions of documents at the same time [5].

As mentioned previously above, the aim of topic modeling is to scrutinize extensive sets of unread text, striving to obtain useful observations into the themes and primary topics of the textual document. For example, it can be exploited to understand motifs behind people opting for a vegetarian or vegan lifestyle. Topic modeling has numerous applications in diverse areas such as marketing, recommendations, information extraction, exploring unfamiliar data, and summing up large text collections, just to cite some. This thesis will delve into the complexities of topic modeling, elaborating on its definition, its numerous applications, and most importantly, its effective employment in analyzing service desk tickets. But first, before diving into these aspects, the dissertation will give an historical context, tracking its origins and discovering how statistical methods have evolved into machine learning models.

2.2 History of Natural Language Processing

Natural Language Processing (NLP) can be described as the pivotal point between artificial intelligence, linguistics and mathematics, with a primary focus of translating human (or natural) language into commands that can be read and executed by computers. NLP consists of two research directions: Natural Language Understanding (NLU) and Natural Language Generation (NLG). The principal mission of NLU is to comprehend the natural language (human language), by deciphering documents and extracting valuable information for downstream tasks. In contrast, NLG is the production of text in natural languages that are understandable by humans based on the provision of structured data, text, graphics, audio, and video.

Natural language, for example text and speech, is defined as unstructured data, meaning it cannot be fit in a relational database. Thus, making it harder to analyze and use for machines.

Natural Language Processing (NLP) and text mining have gathered increasing importance in today's society for numerous reasons. These instruments are greatly utilized in people's daily interactions with technology, often without even them realising. One of their the most prevalent applications is in text messaging, where spelling and grammar corrections rely on NLP. Enhanced NLP techniques contribute to increasing grealy the accuracy and quality of these corrections. Furthermore, an increasing number of individuals are using NLP for translating texts into different languages and communicating with electronic devices, even more emphasizing the importance of these technologies for our modern society.

Natural Language Processing's (NLP) history spans over more than 70 years [6]. Researchers normally trace back its origins to the 1950s, even though precedent work in this field is also recognized [6][6]. In its historic and early stages, NLP mainly consisted of hard-coded rules. These rules relied upon basic grammar principles and word reduction techniques, such as stemming, which involves reducing words to their base or root form (for example, transforming "swimming" into "swim") [6]]. However, these early algorithms had narrow effectiveness due to their simplicity. Furthermore, the lack of computational power and performance capabilities led to a decline in interest in text mining and NLP for numerous years [6].

In the late 1980s and early 1990s, a significant change shook the field of Natural Language Processing [6]. This period was marked by the beginning of the statistical revolution, replacing the inflexible set of hard-coded rules with mathematical and statistical approaches. Methodologies like Latent Semantic Indexing (LSI), probabilistic Latent Semantic Indexing (pLSI), Latent Dirichlet Allocation, and Hidden Markov Models (HMM), to cite a few, were introduced to improve text mining capabilities [7]. It is worthy of notice that some of these methods had prior applications in different fields before and later were adopted for text mining purposes. Among these techniques, Latent Dirichlet Allocation (LDA) gained favor among researchers. Since LDA is built upon concepts from LSI and pLSI, further elaboration on these methods will be provided in the theory chapter.

Afterwards, during later decades, improvements in computing power, integration with machine learning, and enhanced statistical models brought text mining to unprecedented heights. The breakthrough was thanks to Artificial Neural Networks (ANN), which were applied to Natural Language Processing, leading to significant improvements in text mining results. These refined models made possible better translation capabilities, enhanced context understanding, and improved text generation, just to cite some other functionalities [8]. In recent years, scientists have integrated Graphics Processing Units (GPUs) for parallel computing, enabling an accelerated training of ANNs [9]. This development has facilitated faster analysis of always larger datasets and improved overall performance and computing times.

Taking into consideration the swift evolution of Natural Language Processing and the continuous breakthrough of superior models, the future remains unpredictable. However, the ongoing improvements, coupled with a strong and resourceful open-source community, are likely to yield even more refined results and advanced applications in the upcoming years.

Figure 1 below is a flow chart depicting the steps of NLP.

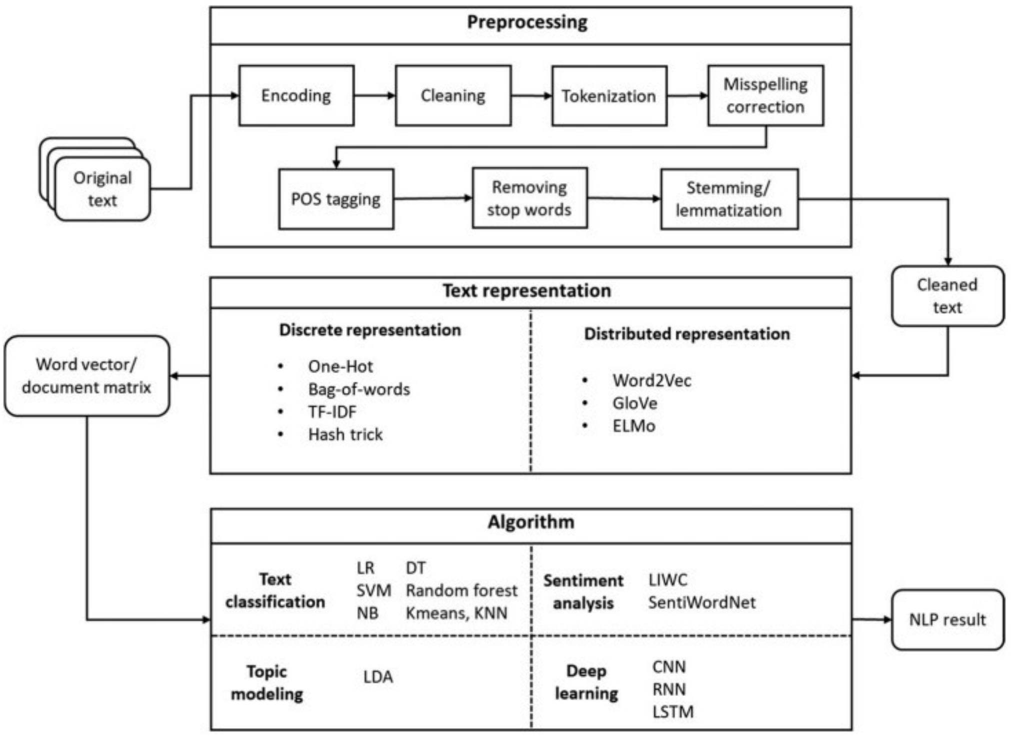


Figure 1: Flow chart of NLP steps. (Yue Kang, Zhao Cai, Chee-Wee Tan, Qian Huang & Hefu Liu (2020) Natural language processing (NLP) in management research: A literature review, Journal of Management Analytics, 7:2, 139-172)

2.3 Current and future use of topic modeling

The evolution of Natural Language Processing over the years has without doubt contributed to the improvement of topic models, which, as we said, heavily rely on NLP techniques. An early prototype of what is now referred to as a topic model was proposed in the early 1990s by Deerwester et al. in their influential paper titled "Indexing by latent semantic analysis" [10].
Approximately a decade later, Blei et al. introduced Latent Dirichlet Allocation (LDA) [11], which has become synonymous with topic modeling for many researchers. LDA is widely preferred due to its consistent delivery of accurate results in most situations [2].

While it might be easy to assume that topic modeling primarily interests researchers only, it does find applications in numerous other fields. One area worthy of note is marketing, where topic modeling has been successfully employed. For example, Amado et al. used topic modeling to identify research trends related to big data in marketing. Their research analyzed 1560 articles published between 2010 and 2015 [12]. By employment of topic modeling, they determined the most dominant dimension among the five factors (big data, marketing, geographic location, sectors, and products) within the articles. This analysis revealed a significant gap between big data research and its application in marketing, providing valuable insights that were previously unknown [12]. This demonstrates the diverse potential applications of topic modeling, stretching beyond the mere research context.

Companies like Bluedot have applied similar methods to detect epidemics and pandemics, as exemplified in their work on detecting Covid-19 [13]. In late December 2019, Bluedot in Toronto, Canada, received a critical warning alert. Their AI-driven algorithm, incorporating machine learning and NLP, detected news resembling the SARS epidemic, gathered from sources in the province around Wuhan, China. Utilizing expert knowledge within the company, they predicted on the same day that a pandemic was likely to hit and anticipated the next cities to be affected [13].

While the specific technology behind Bluedot is undisclosed, it is known that their software relies on AI-driven machine learning models utilizing NLP techniques [13]. Bluedot's software constantly scans information from hundreds of thousands of sources every 15 minutes, 24 hours a day. Their algorithm searches data in over 65 languages, identifying similarities with existing results. The company is keen on emphasizing its reliance on big data and trained models in their approach [14].

Topic models offer the potential to predict and recommend relevant books to individuals as well [5]. By analyzing a person's reading history using topic models, one can discern their preferred books and compare them with other available options. Analogously, in the realm of movies, topic models can analyze what people have watched, compare it with movie reviews and recommendations from others, and generate personalized suggestions [8]. Additionally, topic model recommendations can be applied in this context [8]: models can be utilized not only to understand the texts being analyzed but also to gain insights about individuals themselves [5]. For example, analyzing the books read by historical figures like Charles Darwin provides valuable information about their learning process, thoughts, and exploration of new ideas. This approach can improve the understanding of the person in question [5].

Topic modeling serves as an indispensable tool for identifying new product ideas by analyzing large volumes of product reviews. Through this approach, opinions on innovative products, design concepts, and emerging technologies can be detected. For instance, Ko et al. conducted research on identifying product opportunities using social media mining, where they employed topic modeling to discern product-related discussions among customers [15]. In their study, they utilized topic modeling to analyze extensive product reviews and applied chance discovery theory to generate new product opportunities [15]. While chance discovery theory is beyond the scope of this thesis, further details can be found in their article "Identifying Product Opportunities Using Social Media Mining: Application of Topic Modeling and Chance Discovery Theory" [15].

According to Ko et al., their study provides an expert tool for generating practical product opportunities and enabling product developers to ease up their tasks. By monitoring customer interests and trends, developers can identify emerging topics without the need to manually analyze individual reviews and often not even having to run surveys to clients. This approach proves particularly beneficial for professionals like car designers, allowing them to stay tuned to customer preferences and potential market demands [15]. Ko et al. discovered that rare topics could uncover new product opportunities, whereas frequent topics, although important, do not serve the purpose of identifying new opportunities [15].

The examples above illustrate how topic modeling is not only utilized by researchers but also by various businesses for diverse applications. This trend suggests that the general public is likely to increasingly rely on the capabilities of topic modeling and NLP in the near future. The overwhelming volume of unstructured data, coupled with the continuous influx of new unstructured data published every minute, underscores the need for ongoing advancements in topic modeling [3]. Major companies like BlueDot, Alibaba, Facebook, Apple, and Google, among others, are actively working on improving text mining technologies and exploring new areas where these advancements can enhance people's lives [16].

Dr. Kamran Khan, the founder of BlueDot, imagines that future improvements will enable even earlier detection of new diseases [13]. However, he emphasizes gravely that artificial intelligence was never meant to replace human intelligence but rather to assist in identifying crucial information effectively. As Dr. Khan states, AI facilitates enhanced "needle searches" by pinpointing valuable insights within vast datasets [14]. In addition to the efforts of major companies, the substantial amount of untapped information in the form of unstructured data represents a significant opportunity for future applications [3].

2.4 Related work utilizing topic modeling algorithms.

While topic modeling finds widespread applications in businesses, it remains still a highly researched and utilized method among researchers. In the academic realm, scholars have uncovered similar techniques in their studies, witnessing significant improvements over the years. However, the complicacies of these different methods fall beyond the scope of this study, and therefore, they will not be detailed further in this thesis.

For example, McCallum et al. introduced a probabilistic generative model that concurrently identifies groups among entities and topics among corresponding texts [17]. Their approach, employing a group-topic model, diverges from traditional methods like Latent Dirichlet Allocation (LDA). Unlike pure word distributions, this model clusters entities into groups and words into topics, enabling the discovery of pertinent topics in social networks that might not be discoverable and evident only through word distribution methods [17].

In another study, Chang et al. introduced a novel probabilistic topic model called Nubbi (Networks Uncovered By Bayesian Inference) [18]. Their objective was to utilize topic modeling to deduct relationships among diverse entities like individuals, locations, genes, and companies [18]. Using Nubbi, they successfully identified and comprehended unvelied relationships within plain text in network data, shedding light on the connections among these entities [18].

Additionally, Davison and Hong conducted a study employing topic modeling on Twitter data [19]. They faced the challenge of applying standard topic models in micro-blogging environments like Twitter. The researchers proposed various techniques to train a standard topic model and found that the document length used for training significantly impacts the model's effectiveness. Their research indicated that aggregating short messages often improves training and enhances the quality of the final model, demonstrating the utility of topic modeling approaches in analyzing small and concise texts [19].

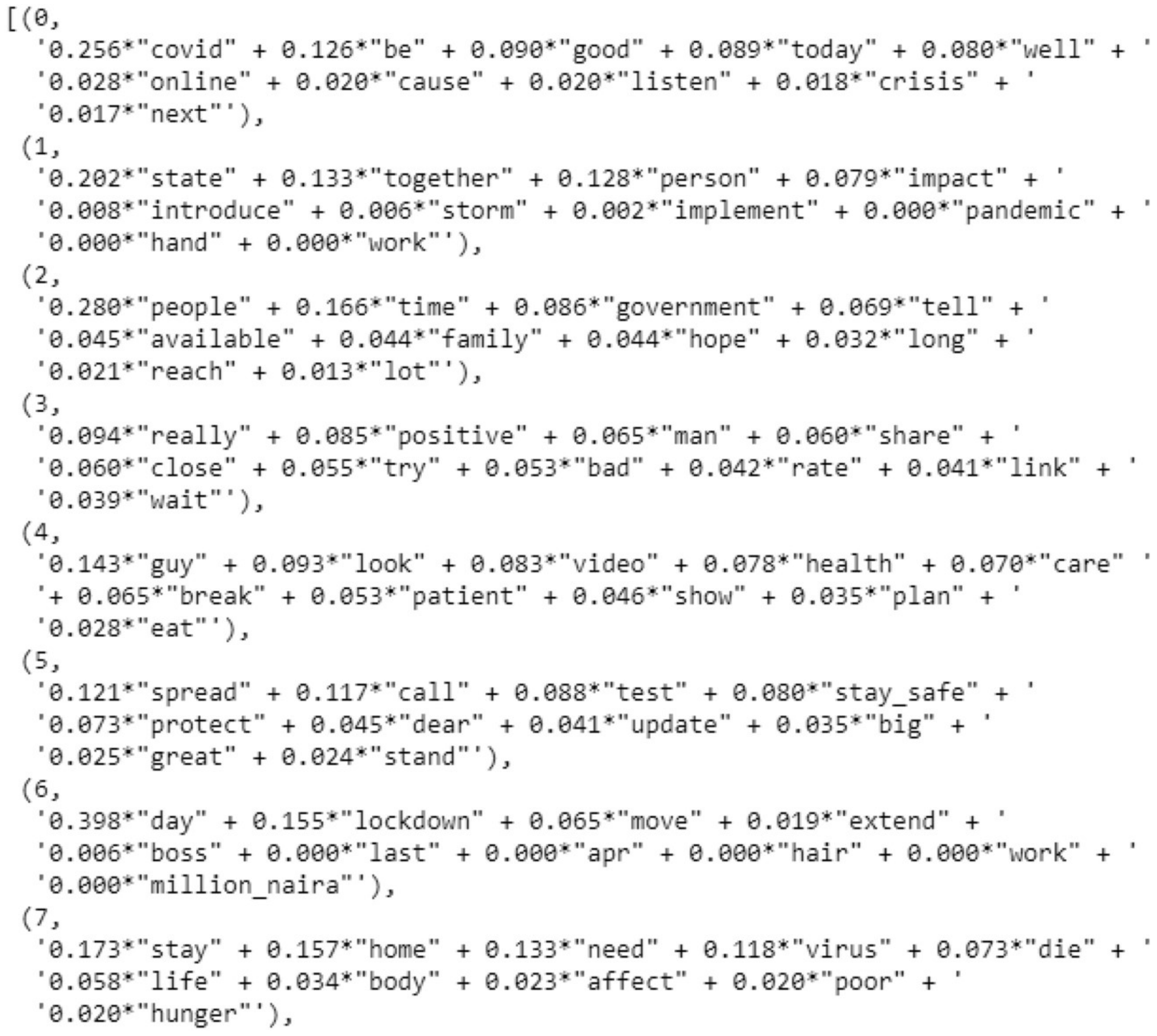
The examples mentioned above highlight a few studies where topic modeling has been utilized to uncover social network groups, establish relationships among various entities, and enhance understanding of micro-blogging environments. With the rising popularity of social networks and micro-blogging platforms, the ability to automatically understand and analyze these platforms is becoming increasingly vital. As emphasized in the aforementioned studies, significant portions of the information discovered through these models would have been unlikely to be detected using methods other than topic modeling onesn[17][26][18][19].

3 Theory

Like previously mentioned in the introduction and background, numerous techniques and tools are available for performing topic modeling. However, before diving into the intricate statistics and mathematical staples of these methods, it is essential to establish a fundamental intuition about the algorithms in general and comprehend the type of results they output. As previously explained, topic modeling serves the purpose of exploring extensive sets of text data, aiming to provide an overview of the essential aspects necessary for summarizing the texts. Maybe contrary to expectations, the results of topic modeling often manifest as arrays of words accompanied by numerical values, rather than complete or partially complete sentences. This format is exemplified in Figure 3: simple tokenization performed by a Python built-in function., derived from a sentiment analysis on Covid-19 in Nigeria.

At the start of each inner list, the numbers 0 through 9 represent the ten distinct topics generated through topic modeling. Within the inner lists, numbers like 0.256*”covid”, signify the weights, or probabilities, indicating the likelihood of that specific word being associated with the corresponding topic. These lists are organized in a descending order, with the first word being the most likely to appear within a particular topic. The length of the outer list is determined by the user, who arbitrarily selects the number of topics *k*. Similarly, the user can sometimes also decide the number of words to be displayed per topic, shaping up the inner lists.

From each inner list, users can deduct their own understanding of the theme and assign a topic name based on the words provided automatically by the script. It's important to note that the words within these inner lists are drawn from the vocabulary created from the corpus, encapsulating all words within the analyzed text dataset. However, as mentioned earlier, only words with the highest probabilities are displayed.



```
[(0,
  '0.256*"covid" + 0.126*"be" + 0.090*"good" + 0.089*"today" + 0.080*"well" + '
  '0.028*"online" + 0.020*"cause" + 0.020*"listen" + 0.018*"crisis" + '
  '0.017*"next"'),
 (1,
  '0.202*"state" + 0.133*"together" + 0.128*"person" + 0.079*"impact" + '
  '0.008*"introduce" + 0.006*"storm" + 0.002*"implement" + 0.000*"pandemic" + '
  '0.000*"hand" + 0.000*"work"'),
 (2,
  '0.280*"people" + 0.166*"time" + 0.086*"government" + 0.069*"tell" + '
  '0.045*"available" + 0.044*"family" + 0.044*"hope" + 0.032*"long" + '
  '0.021*"reach" + 0.013*"lot"'),
 (3,
  '0.094*"really" + 0.085*"positive" + 0.065*"man" + 0.060*"share" + '
  '0.060*"close" + 0.055*"try" + 0.053*"bad" + 0.042*"rate" + 0.041*"link" + '
  '0.039*"wait"'),
 (4,
  '0.143*"guy" + 0.093*"look" + 0.083*"video" + 0.078*"health" + 0.070*"care" '
  '+ 0.065*"break" + 0.053*"patient" + 0.046*"show" + 0.035*"plan" + '
  '0.028*"eat"'),
 (5,
  '0.121*"spread" + 0.117*"call" + 0.088*"test" + 0.080*"stay_safe" + '
  '0.073*"protect" + 0.045*"dear" + 0.041*"update" + 0.035*"big" + '
  '0.025*"great" + 0.024*"stand"'),
 (6,
  '0.398*"day" + 0.155*"lockdown" + 0.065*"move" + 0.019*"extend" + '
  '0.006*"boss" + 0.000*"last" + 0.000*"apr" + 0.000*"hair" + 0.000*"work" + '
  '0.000*"million_naira"'),
 (7,
  '0.173*"stay" + 0.157*"home" + 0.133*"need" + 0.118*"virus" + 0.073*"die" + '
  '0.058*"life" + 0.034*"body" + 0.023*"affect" + 0.020*"poor" + '
  '0.020*"hunger"'),
```

Figure 2: example result of topic modeling (Abiola, O., Abayomi-Alli, A., Tale, O.A. et al. Sentiment analysis of COVID-19 tweets from selected hashtags in Nigeria using VADER and Text Blob analyser. Journal of Electrical Systems and Inf Technol 10, 5

During the process of performing topic modeling with LDA, two matrices are obtained: the word-to-topic probability matrix and the document-to-topic probability matrix [20]. These two outputs, often given in .xlsx format, explain the likelihood of a word being associated with a topic and the likelihood of a document being associated with a topic.

Thus, topic modeling is a form of cluster analysis, it discerns similarities among different words. Visualizations created using the pyLDAvis software package, provide visual insights into why topic modeling functions as a cluster analysis tool, but for the scope of this thesis, graphical visualization was obtained through the use of the software Dariah, as mentioned in the Introduction.

Throughout the thesis, specific terminology is consistently used, and it is crucial to interpret it accurately. A corpus refers to a collection of documents, whereas these documents are composed of words, also known as tokens, which are items from a vocabulary made of up of words present in the corpus. This vocabulary is unique to each analysis, tailored based on the documents analyzed.

3.1 Preprocessing

In order to conduct analyses on a dataset, preprocessing is a crucial step. Preprocessing serves as a "cleaning" process for the data, aiming to eliminate unnecessary disruptions like stop-words, punctuation, and extraneous symbols, among others. There are numerous ways to clean up text data, and it is crucial to focus on the specific objective. Computers require text to be converted into numbers, as they cannot comprehend human-readable text [20]. The goal of this thesis was to train a topic model and assess how effective the exploration of unread text data, in the form on service desk tickets, could be. For this reason, this section will cover some of the most commonly used preprocessing steps in topic modeling. However, not all the preprocessing steps outlined below were employed in this thesis. This section serves to identify suitable preprocessing methods, ensuring that the decisions made had a theoretical ground.

As previously discussed in the background section, texts come in various lengths and topics can vary heavily in the number of available documents. For instance, there are millions of documents about politics, while only a limited number are available on specific topics like Shakespeare's plays [21]. This diversity poses challenges for the algorithms. Tang et al. noted in their study that short documents or a scarcity of documents about a particular topic can limit the quality of results in topic modeling [21]. Analyzing entire books can also be challenging due to the multitude of topics they encompass [21]. Misspelled words, abbreviations, and language variations inside a collection can complicate topic understanding and human evaluation. However, these issues can be addressed through techniques such as term frequency-inverse document frequency (tf-idf) or vocabulary reduction by imposing restrictions on the occurrence of words within documents and across the entirety of the corpus [22].

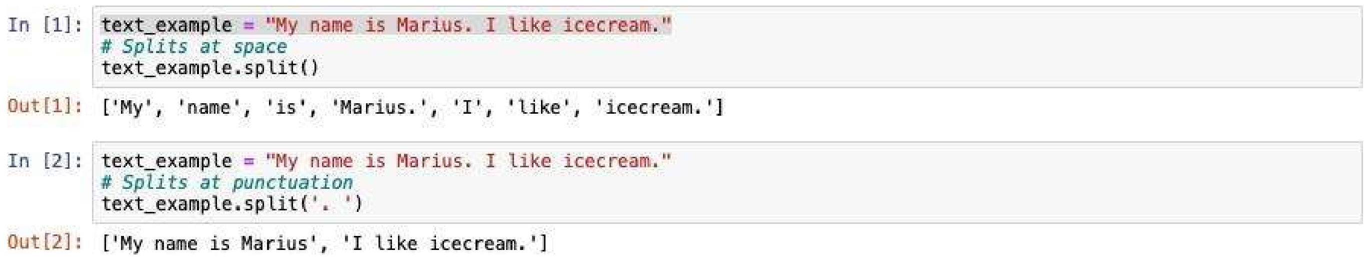
It is important to acknowledge that the order of the following theoretical section corresponds to the sequence of preprocessing steps within this thesis. However, not all studies on topic modeling often agree on the order of these steps.

3.1.1 Tokenization

Tokenization serves as the initial step in preprocessing for most text mining tasks. This process involves dividing an entire corpus into documents, which are then further split into individual words, terms, or symbols, known as tokens [23]. Tokens are typically separated by white space or full stops, meaning continuous strings of letters of the alphabet are identified as distinct tokens. During tokenization, there are various methods to enhance the quality and accuracy of tokens. These methods include restricting token length, eliminating unnecessary tokens such as punctuation, and converting all tokens to lowercase, to cite a few. These steps are needed to simplify the corpus and enhance the likelihood of obtaining valid results [23].

For the context of topic modeling, it is crucial to remove tokens consisting solely of punctuation as they can adversely affect the resulting topics. For instance, including punctuation characters within the inner lists of topics can distort the interpretation, while also skewing results, as punctuation is very frequent inside text. Additionally, converting all words to lowercase is essential because failure to do so would lead the computer to differentiate between identical words with different cases. For example, "cat," "Cat," or "CAT" would be treated as distinct words if not converted to lowercase [23].

Possibly the simplest method of tokenization is to use a built-in function in Python such as: *.split()*. This method splits words based on the spaces in each document or alternatively a predefined splitting parameter. This is shown in Figure 3.



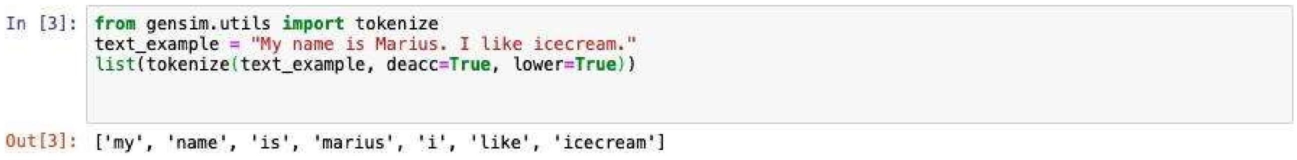
```
In [1]: text_example = "My name is Marius. I like icecream."
        # Splits at space
        text_example.split()

Out[1]: ['My', 'name', 'is', 'Marius.', 'I', 'like', 'icecream.']

In [2]: text_example = "My name is Marius. I like icecream."
        # Splits at punctuation
        text_example.split('. ')

Out[2]: ['My name is Marius', 'I like icecream.']
```

Figure 3: simple tokenization performed by a Python built-in function.

As seen in the figure, the top example is two sentences split based on white space, while the lower example is a split based on punctuation.

An alternative way of performing tokenization is using a package built in Python. Here there is a number to choose from, such as **NLTK**, **spacy** and **Gensim**. Figure 4 shows how Gensim is applied with respect to the same example as above.

However, it also shows how easy it is to remove punctuation and lower-case all words in the same operation.



```
In [3]: from gensim.utils import tokenize
        text_example = "My name is Marius. I like icecream."
        list(tokenize(text_example, deacc=True, lower=True))

Out[3]: ['my', 'name', 'is', 'marius', 'i', 'like', 'icecream']
```

Figure 4: tokenization by the mean of Gensim

3.1.2 Stemming or Lemmatization

The subsequent step in preprocessing, in most cases, involves either stemming or lemmatization, both of which serve a similar purpose. Stemming is a process that reduces words (tokens) to a common stem form, whereas lemmatization reduces words to their base or dictionary form [24]. Stemming eliminates inflections and derivational suffixes to simplify words into their stems, reducing vocabulary by excluding multiple forms of the same word. However, there is a risk of either under-stemming or over-stemming. Under-stemming retains two words from the same conceptual group differently, while over-stemming combines two words from the same conceptual group into one [24]. For instance, under-stemming might occur with the words "running" and "ran," where both "run" and "ran" are retained within the same conceptual group despite having the same meaning. Over-stemming, on the other hand, might occur with "new" and "news," where both become "new," diminishing two distinct conceptual groups into one. Stemming can be performed using various algorithms, many of which are based on simple rules that remove word endings.

Lemmatization utilizes an external vocabulary (with numerous available options) and conducts a morphological analysis of words within the corpus. In this context, morphological analysis refers to identifying relationships with other words. Based on these relationships, words are reduced to their dictionary forms [24]. Lemmatization also examines the part-of-speech category for each word to identify the correct dictionary form. Part-of-speech categories include nouns, pronouns, verbs, adjectives, and adverbs. Some predefined methods also include checks for synonyms of the same word [24].

3.1.3 Stop-words

Additionally, it is crucial to remove stop-words from the text data. Examples of stop-words include common words such as "I","and","the" and "are". The removal of stop-words significantly impacts the quality of the topics generated during analysis [25]. These words tend to be disproportionately represented in the topics as they are ubiquitous in English sentences [25]. Stop-words are primarily used for sentence structure and readability, providing filler words for the reader. However, it is essential to carefully choose which words to remove, as some removals might inadvertently eliminate important data. Removing certain stop-words could completely alter the meaning of a sentence. For instance, the phrase "not like" becomes "like" after removing stop-words, as "not" is considered a stop-word.

There are two commonly used Python packages for stop-word removal: spacy and NLTK. Both packages remove stop-words based on predefined lists. While there is considerable overlap between the stop-word lists of these packages, some words may be unique to each package.

3.1.4 Conversion and transformation into digits

In most text mining applications, the next step in preprocessing is to convert words or tokens into numerical values suitable for computational analysis. As stated in the book "Python Machine Learning" by Raschka and Mirjalili, text or words need to be transformed into a numerical format before they can be processed by a machine learning algorithm [20], p. 259]. Several methods can accomplish this, including bag-of-words, tf-idf, and word embedding with word2vec. This thesis focuses primarily on the bag-of-words method, as it provides the numerical feature vectors necessary for LDA [20],p. 274].

To implement the bag-of-words approach, one first creates a vocabulary of unique tokens (words) from the entire corpus [20], p. 259]. Then, a feature vector is constructed for each document in the corpus, containing information on the frequency of each word within that specific document [20], p. 259]. These feature vectors are often sparse, mostly comprising 0’s. This sparsity arises due to the size of the corpus vocabulary, which is typically much larger than the vocabulary of any individual document.

To illustrate this process, consider a corpus consisting of 100 documents with a total vocabulary of 1000 unique words. These 1000 words form the corpus vocabulary [20],p. 259]. Next, the words are tabulated, and their frequencies are counted across the entire corpus [20],p. 259]. Binary bag-of-words vectors or one-hot encoding vectors, as explained by Bengfort et al. [26], p. 59], can be created. Binary bag-of-words vectors indicate only the presence or absence of a word within a given document, disregarding the word's frequency. On the other hand, bag-of-words vectors account for the frequency of each word within each document [20] p. 259]. Both binary and bag-of-words vectors are constructed so that each number within a vector represents the existence (set to 1) or frequency of a word from the vocabulary within a specific document. If the word does not exist within the document, its value is set to 0 [20], p. 259]. Consequently, these vectors are mostly sparse, with mostly 0’s and a few 1’s (or frequency counts) representing words present in the document.

Due to the sparsity of bag-of-words, this method consumes a significant amount of memory, especially when dealing with large corpora [20], p. 259]. This can pose problems for computers with limited memory capacity. Additionally, bag-of-words vectors treat all words equally, disregarding the importance of specific words within the documents [20].To address this issue, term frequency-inverse document frequency (tf-idf) can be employed. Tf-idf allows certain words to be assigned higher weights or to be completely excluded if they are deemed unimportant.

As introduced earlier, an alternative to bag-of-words and tf-idf is word embedding. In word embedding, each word is mapped to a unique vector representation, aiming to preserve semantic similarities by assigning similar vectors to related words. One prominent word embedding method is word2vec, an algorithm introduced by Google in 2013 as an alternative to the bag-of-words approach [[20], p. 274]. Word2vec is an unsupervised algorithm based on neural networks, aiming to learn relationships between words [20], p. 274]. It clusters words with similar meanings into similar vector spaces using vector math [27]. Word2vec effectively captures the semantic relationships between words. For instance, words like "cat" and "tiger" would likely have more similar vector representations compared to "cat" and "computer."

Word2vec operates with two different architectures, both serving a similar purpose [27]. Both models aim to create word vectors, but they differ in their approaches to training the model for vector creation. The first model is the continuous bag-of-words (CBOW) model, where the order of words does not influence vector creation [27]. CBOW employs a neural network with hidden layers to predict the target word. More detailed information can be found in the paper "Efficient Estimation of Word Representations in Vector Space" by Mikolov et al., the creators of this method [27].

Although the mathematical details are not explained here, there exists a simplified representation of the model, as shown in Figure 5. The figure illustrates both CBOW and skip-gram and highlights their differences. In CBOW, surrounding words serve as input, meaning words appearing directly before and after the word to be predicted. According to Mikolov et al., CBOW utilizes four preceding and four subsequent words of the target word for prediction [27]. In contrast, skip-gram operates in reverse, taking an existing word as input and predicting the surrounding words. Both models do not show the hidden neural network layers, which are essential components of the actual models.

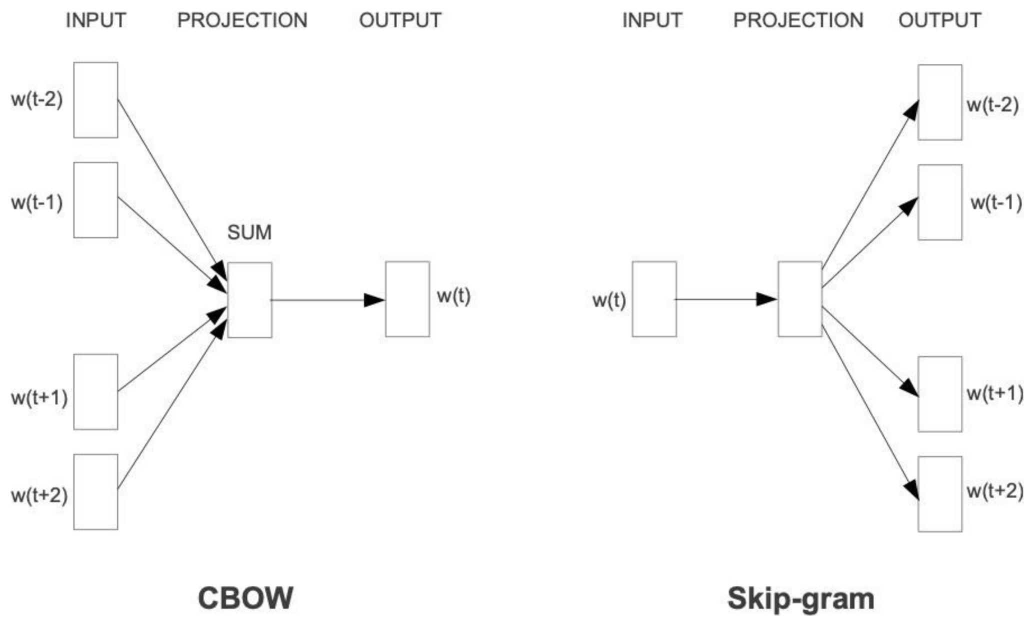


Figure 5: Illustration of CBOW and Skip-gram models (Mikolov Tomas, Chen Kai, Corrado Greg, Dean Jeffrey, (2013) "Efficient Estimation of Word)

3.2 Topic modeling

The subsequent stage involves the analysis and generation of topics from the preprocessed and cleaned data. Various methods exist for creating topics, and among them, Latent Semantic Indexing (LSI) and Latent Dirichlet Allocation (LDA) are prominent models widely employed in both research and industry. While discussing the fundamental concepts and providing some mathematical insights, the primary focus will be on LDA, as it is the more preferred and extensively used model [2].

The primary objective of any machine learning model is to achieve the best fit for the data, and this holds true for a topic model. In this context, the aim is to comprehensively explore and provide an overview of the content discussed in the documents within the corpus. The goal is to extract the maximum information available, emphasizing the need to cover all aspects of the corpus. While an ideal automatic measure that gauges the information received and the human interpretability of topics would be beneficial, such a metric does not currently exist. Researchers are actively working on identifying the most effective method.

Alternative approaches to measure the interpretability of topic models include word intrusion and coherence. Word intrusion requires human intervention to identify non-related words, making it a labor-intensive process that demands expert knowledge within the specific research field. Coherence, in contrast, is an automated process that identifies semantic similarities among words across topics. Both methods will be elucidated further after the sub-sections covering LSI and LDA.

3.2.1 Latent Semantic Indexing

Latent Semantic Analysis (LSA) was developed in the early 1990s with the goal of enhancing the detection of relevant documents based on simple search queries [13]. The method focused on uncovering underlying latent semantic structures through statistical techniques. Deerwester et al. applied singular-value decomposition (SVD) to a matrix of terms and documents, such as bag-of-words, to construct a "semantic" space where closely associated terms and documents were positioned together. SVD enabled the elimination of less influential influences while highlighting significant associative patterns, including word similarities. Additionally, it led to the identification of less crucial terms to a given document, making them appear close to that same document due to the major patterns within the rest of the data. These new spatial positions were then utilized to identify topics within the dataset [10].

For a more in-depth understanding of how LSA operates, a slightly more mathematical explanation is necessary. The method begins with a matrix of documents by terms (words), which serves as the input matrix. The creation of such a matrix can involve the use of the bag-of-words method [10]. In this approach, the frequency of each word in the vocabulary is tallied per document. Alternatively, a binary bag-of-words approach can be employed, where the existence of a word is denoted as 1, while 0 indicates the absence of the word within the document.

Furthermore, this matrix undergoes analysis using Singular Value Decomposition (SVD) to unveil the latent semantic structure and reduce the dimensionality of the input data [10]. Equation 2.3 illustrates the definition of SVD. Matrix $\boldsymbol{A}$ represents the input matrix of m documents and n terms. Matrix $\boldsymbol{U}$ comprises the left singular vectors of m documents by r topics. Matrix $\boldsymbol{\Sigma}$ is the diagonal matrix of singular values with dimensions $r \ x \ r$,, where r is the rank of matrix $\boldsymbol{A}$. This matrix signifies the strength of each topic, with diagonal values sorted from largest to smallest. The diagonal matrix has all values set to zero except on the diagonal. Matrix V^T consists of the right singular vectors of r topics by n terms [10]. It's crucial to note that U and V are orthogonal to each other.

$$A_{[m*n]} = U_{[m*r]}\Sigma_{[r*r]}(V_{[n*r]})^T \tag{2.3}$$

Truncated SVD is subsequently employed, meaning only the vectors of U related to the largest values of Σ are retained, while the rest are discarded [10]. The simplified version of Equation 2.3 becomes 2.4. Here, $\hat{A}$ represents an approximate decomposition that closely aligns with the actual A. The approximate $\hat{A}$ has a rank of t topics, replacing each instance of r with t as the new rank. In other words, setting the number of topics truncates the SVD, providing an approximate decomposition of the input matrix A. As previously mentioned, determining the appropriate number of topics is somewhat intricate, as one aims for a topic number large enough to accommodate all real data, yet small enough to exclude unimportant details.

$$A \approx \hat{A} = U_t \hat{\Sigma}_t (V_t)^T \qquad (2.4)$$

The actual matrix calculations and more intricate details regarding interpretation techniques are beyond the scope of this thesis. Nevertheless, it is crucial to recognize that the decomposition can be utilized to further explore the topics. The U matrix reveals the relationship between each document and the topics, while the V^T matrix elucidates the relationship between the words (vocabulary) and the topics.

In 1999, Thomas Hofmann introduced a novel approach to LSI called Probabilistic Latent Semantic Indexing (pLSI) [28]. Hofmann asserted that his method boasted an improved statistical foundation, grounded in the likelihood principle and a proper generative model. While these aspects won't be elaborated on here, he explicitly highlighted that his model facilitated an understanding of polysemy by distinguishing between meanings and different word usages. Polysemy refers to the phenomenon where the same word, phrase, or symbol can have several distinct meanings, such as "*get*," which can mean "*become*," "*procure*," or "*understand*." Additional details about pLSI can be found in the work titled "Probabilistic Latent Semantic Indexing" [28].

3.2.2 Latent Dirichlet Allocation

Latent Dirichlet Allocation (LDA) is described as "a generative probabilistic model for collections of discrete data such as text corpora" [11]. Proposed in 2003, LDA builds on the foundational principles of pLSI, introduced a few years earlier by Hofmann [11]. LDA is a clustering method that aims to group frequently co-occurring words across a corpus [20], p. 275], akin to fuzzy K-means clustering. Unlike traditional K-means, which assigns each observation to a single cluster based on the nearest mean, fuzzy K-means allows an observation to belong to multiple clusters. LDA is defined as a mixed membership model, which is why it shares this characteristic with fuzzy K-means and not just K-means.

Inherently, LDA is an unsupervised machine learning method, inferring patterns without reliance on known outcomes or labels. Unlike supervised machine learning, where the model is trained on known labels, LDA is suited for exploring unknown datasets, offering meaningful insights into patterns and structures. While supervised variants like Labeled-LDA [29] and supervised-LDA [30] exist, they won't be delved into further.

Before delving into the mathematical intricacies of LDA, it's crucial to grasp the basic concept. As mentioned earlier, LDA enables the explanation of unobserved groups of similar data through a set of words known as topics. In this method, documents are taken as input, forming a corpus. This corpus undergoes preprocessing before being fed into the LDA model in the form of a bag-of-words. The model subsequently outputs a list of lists of words representing the identified topics, as illustrated in Figure 2.

Figure 1Figure 6 provides a well-known illustration of how topics are generated, created by David Blei, one of the pioneers of LDA for topic modeling [5]. In this depiction, each document, represented by the text document in the middle, encompasses multiple topics, each identified with distinct colors. The colors serve to distinguish different "topics." If we envision coloring all words and excluding stop-words, it becomes feasible to discern the overall themes (or topics) simply by reading a few words of each color. LDA operationalizes this intuition by constructing a probabilistic model of text [5].

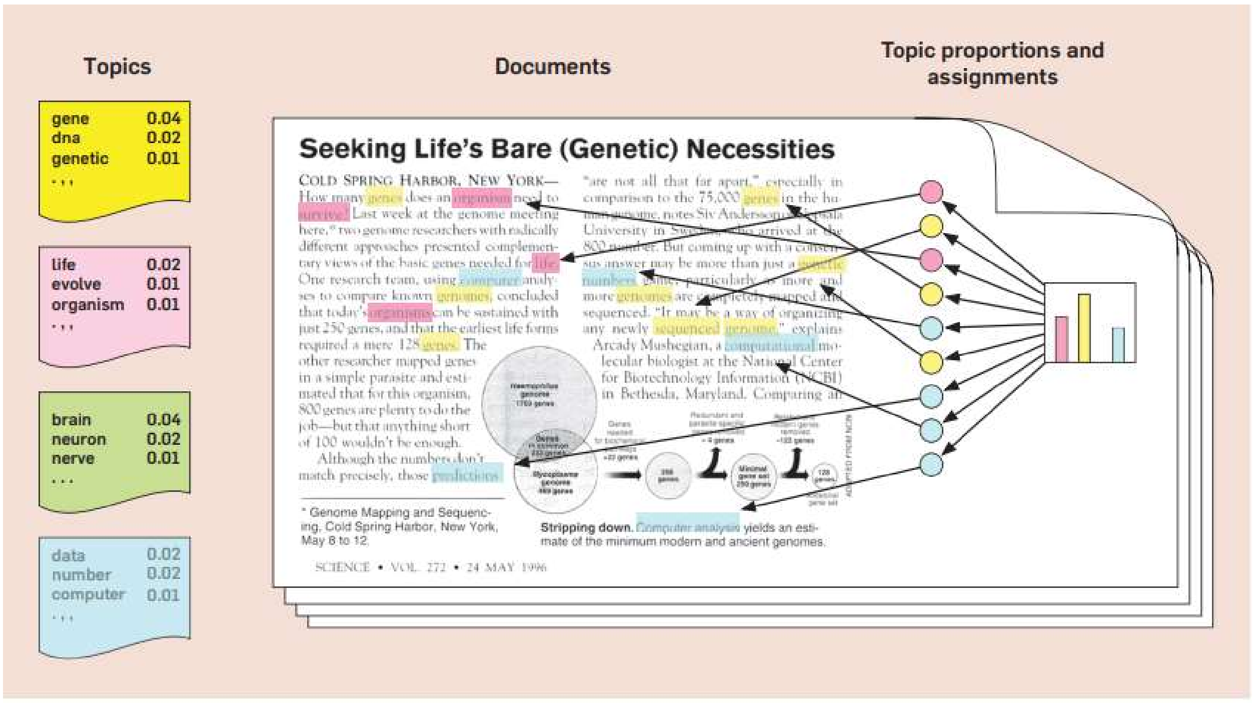


Figure 6: LDA generation process. The figure shows a known illustrative example of how topics are found by using LDA. The left most boxes are the actual topics, while the text is an example of a document within the corpus. The colored dots and the hist

Examining the entire figure provides a clearer understanding of the LDA sequence. The topics displayed on the left represent distributions of words derived from the vocabulary established from the corpus. The user fixes and sets the number of topics. Each number alongside the words within the topics signifies the probability of that word belonging to the given topic.

LDA assumes that each document is generated in the following way: initially, the model selects a random distribution over topics, illustrated as the histogram on the far right. Subsequently, for each word within the document, a color is chosen to identify the closest interpretation of the word (i.e., the topic to be placed within). This process is iterated for every word in the document, constructing the document itself. The entire process is repeated multiple times, where each word within a given document is assigned to a new distribution over topics. Throughout this process, the number of topics remains constant, while the degree to which a document exhibits a given topic changes. This model is referred to as a mixed-membership model, explaining its similarity to fuzzy K-means and not just K-means. A given topic contributes to the construction of several documents, and a word contributes to the construction of several topics.

Latent Dirichlet Allocation can be elucidated further from a mathematical perspective. LDA identifies a collection of words representing topics, assuming that a corpus comprises documents composed of various words. The method uses bag-of-words as the vector representation of the corpus and produces a document-to-topic matrix as well as a word-to-topic matrix [20], p. 275]. Bag-of-words neglects the order of words in a document, known as the assumption of exchangeability [11]. This assumption holds across documents within the corpus; the order of documents can be disregarded [11]. These assumptions form the basis of the LDA model described in this thesis: "mixture models that capture the exchangeability of both words and documents"[11].

To facilitate a deeper understanding of the mathematics behind the model, a graphical model is beneficial. Figure 7 illustrates the different components of the model. The nodes represent random variables, with the shaded one being observed, while the unshaded ones are hidden [5]. Edges indicate dependencies, and plates signify replicated variables [5]. Replicated variables denote repeated entities [5]. The outer plate represents the documents, while the inner plate represents the words and their positions within the documents. To provide more specificity and establish a connection to the illustration in Figure 6, the Dirichlet parameter and the Per-document topic proportions are depicted as the histogram. In each iteration of training, every word within the text is assigned to one of the topics. The Per-word topic assignment is portrayed as colored buttons, signifying the assignment of each word to the best-fitting topic. The Observed word is the actual text document that has undergone preprocessing. The combination of Per-topic word distributions and the Topic parameter constitutes the list of topics displayed on the far left. These topics are utilized in each iteration when extracting a distribution of topics.

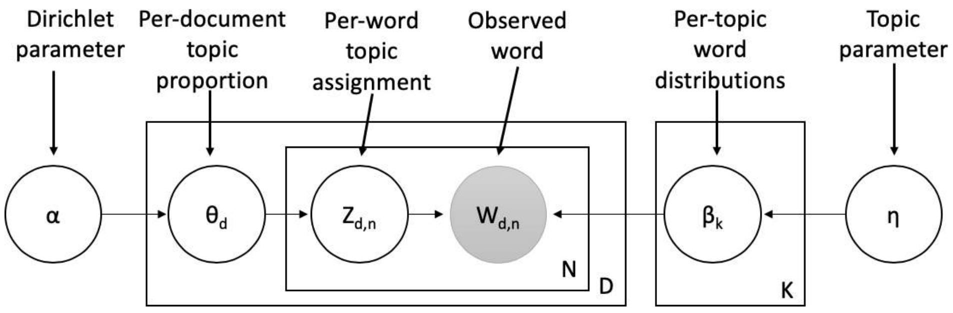

Dirichlet parameter
Per-document topic proportion
Per-word topic assignment
Observed word
Per-topic word distributions
Topic parameter
α
θd
Zd,n
Wd,n
βk
η
N
D
K

Figure 7: Graphical representation of LDA: The depicted model serves as a faithful reproduction of the graphical model introduced by David Blei during his guest lecture at the University of Edinburgh. This model elucidates the components involved in the generation of topics.

The Greek and Roman letters in the model serve distinct purposes, each contributing to its functionality [5]:

- D represents the total number of documents in the corpus.

- N is the number of words within a specific document.

- K denotes the number of topics, a value set by the user.

- α is the Dirichlet parameter vector influencing the distribution over topics chosen for each iteration. A large value results in an even distribution, while a small value favors certain topics.

- η is the Dirichlet topic parameter vector, representing the Dirichlet distribution influencing the distribution of words within each topic. Similar to

- Wd,n, n is the observed words within the documents. More specifically it is the nth word in the sequence within the dth document, where n goes from 1 to N and d from 1 to D.

- Zd,n is per-word topic assignment of each of the nth word within the dth document.

- βk consists of vectors representing each of the kth topic, where k goes from 1 to K. In other words, each vector βk is the distribution over the full vocabulary and this sums up to 1. These vectors are put together into a matrix which constitutes the word-to-topic probability matrix, an output from an LDA model.

- θd represents vectors signifying the per-document topic proportions for each of the dth documens, where d ranges from 1 to D. It essentially indicates the likelihood of a given topic for each document in the corpus. The amalgamation of these vectors generates the document-to-topic probability matrix, a second output from an LDA model.

The graphical model aims to estimate the hidden variables by computing their distributions based on the given documents. The objective is to estimate the probability of topics, proportions, and assignments given the words, denoted as P (topics, proportions, assignments | words). This is achieved by initially selecting a distribution θd based on the Dirichlet distribution α. The Dirichlet distribution is a vector that always sums up to 1 and is utilized to elucidate the probability distribution of the θ probability distribution [11]. The Dirichlet distribution itself is beyond the scope of this thesis. However, the impact of different α values is significant. A small value implies that the document will have a few prominent topics, while a high α value implies that the documents are a mixture of most topics [11].

Furthermore, a topic zd,n is assigned from the θd for each of N words within the given document. This is a latent variable. Finally, the words wd,n are sampled and conditioned on the zd,n topic. In other words, θd explains the probability to which a topic i explains a given document d. The mathematical details are beyond the scope of this thesis.

The symbol **β** represents the topics and is a matrix created from the $\boldsymbol{\beta_k}$ vectors, where each row is a unique $\boldsymbol{\beta_k}$ vector. Within the **β** matrix, each row represents a topic, and each column is a word within the topic, as shown in Table 1. The values within the matrix indicate the probability of a topic *i* containing the word *j*. Each $\boldsymbol{\beta_k}$ vector is the distribution over the full vocabulary, summing up to 1. Larger values indicate a higher probability of the specific word being included in a given topic, while a zero value indicates that the word is not likely to have importance within the topic. The words are usually evenly distributed, but this distribution can be adjusted to favor certain words of a particular topic [11]. This distribution is based on the η, which is a Dirichlet distribution parameter similar to α. The only difference is that this affects the word distribution instead of the topic distribution, and the same technique can be used here [11]. A low value will favor just a few words per topic, while a high value will result in an even distribution. Both the *η* and the *α* values can be set manually. It is, however, only needed when there exists previous knowledge about the word and/or topic distributions, respectively.

Word probabilities for each topic							
	Word 1	Word 2	Word 3	Word 4	Word 5	Word 6	Word 7
Topic 1	0	0.1	0.5	0	0.3	0.05	0.05
Topic 2	0.7	0.05	0.05	0.05	0.05	0	0.1
Topic 3	0.3	0.3	0.2	0.1	0.03	0.07	0

Table 1: β matrix for LDA. This is a visualization of the β matrix which is the result when performing LDA. Each of the $\boldsymbol{\beta}_k$ vectors put together creates this matrix. The result is usually visualized as a list of lists instead, similar to Figure 2

Based on the information provided, it is evident that, given a set of documents, a vocabulary is established. Subsequently, Latent Dirichlet Allocation (LDA) is employed to generate a *θ* document-to-topic probability matrix, representing the distribution of topics per document, and a *β* word-to-topic probability matrix, representing the distribution of words per topic. These matrices can then be analyzed to delve into the content of the given documents, both at the individual document level and across all documents. However, a challenge arises in assessing the quality of the identified topics and the extent to which these topics encapsulate the themes addressed within the documents. This challenge is closely tied to determining the optimal number of topics.

4 Research objectives and research framework

The primary goal of this study is to delve into the potential of topic modeling algorithms, specifically Latent Dirichlet Allocation (LDA), in the context of automatic and unassisted categorization of service desk tickets. This research is situated at the intersection of Natural Language Processing (NLP) and service desk operations, aiming to bring about significant improvements in the efficiency and velocity of ticket categorization.

The specific objectives of this research are as follows:

Feasibility Assessment:

The first objective is to investigate the feasibility of the LDA algorithm for capturing underlying themes and topics within service desk requests. This involves an objective assessment of the algorithm's ability to handle the unstructured nature of ticket data, enabling the extraction of meaningful patterns from diverse and complex textual information.

Model Development:

The second objective is to develop an optimal model through multiple iterations. This involves empirically determining parameters such as the optimal number of topics and iterations, as well as addressing issues related to corrupted data and outliers. The aim is to fine-tune the model to ensure it accurately captures the inherent structure and themes within the service desk tickets.

Enhancing Interpretability:

The third objective is to explore methods to enhance the interpretability of topics. This involves ensuring that the generated categories align with domain-specific knowledge of service desk operations. The goal is to produce topics that are not only statistically significant but also meaningful and interpretable from a service desk perspective.

Performance Evaluation:

The fourth objective is to evaluate the performance of the LDA-based ticket categorization model in terms of accuracy and coherence. This involves comparing the model’s performance against manual categorization benchmarks, providing a rigorous assessment of its effectiveness.

Scalability Analysis:

The final objective is to analyze the scalability of the LDA model concerning large volumes of service desk tickets. This involves testing the model’s performance in high-throughput environments, ensuring its applicability in real-world scenarios.

By addressing these objectives, this research aims to contribute valuable insights and practical solutions to the field of service desk ticket management. It seeks to bridge the gap between topic modeling techniques and service desk operations, potentially revolutionizing how service desks handle ticket categorization. The outcomes of this study have far-reaching implications, with the potential to significantly enhance efficiency, accuracy, and responsiveness in service desk operations, ultimately benefiting both service providers and end-users alike.

5 Methodology

As mentioned above, this study aims to explore the efficacy of topic modeling algorithms, particularly Latent Dirichlet Allocation (LDA), in automating the categorization of service desk tickets. Positioned at the intersection of Natural Language Processing (NLP) and service desk operations, this research aims to delve into potential substantial enhancements in the efficiency and swiftness of ticket categorization processes.

The research methodology begins with the assembly of a comprehensive corpus of customer support tickets. Subsequently, the dataset undergoes meticulous cleaning and preprocessing to remove non-relevant metrics such as dates, ticket ID's, and user-specific information. This refining process ensures a refined corpus suitable for analysis within topic modeling scripts.

Upon configuring the requisite parameters for the LDA model, the script is executed, generating results. These outcomes are meticulously collected, systematically analyzed, and subjected to rigorous evaluation to derive meaningful insights.

The research methodology encapsulates a systematic progression from data acquisition to preprocessing, modeling configuration, script execution, and final analysis and evaluation. This comprehensive approach allows for a robust exploration of the potential of LDA in streamlining service desk ticket categorization.

5.1 Data collection

During my internship in Lavazza, the service desk data pertaining to the year 2020 was shared with me by the company's AMS (Application Management Services) Manager. The database had undergone aggregation, eliminating any potential privacy concern in the process of utilizing it for topic modeling research.

The data, presented in .xlsx format, exhibited the following structural arrangement:

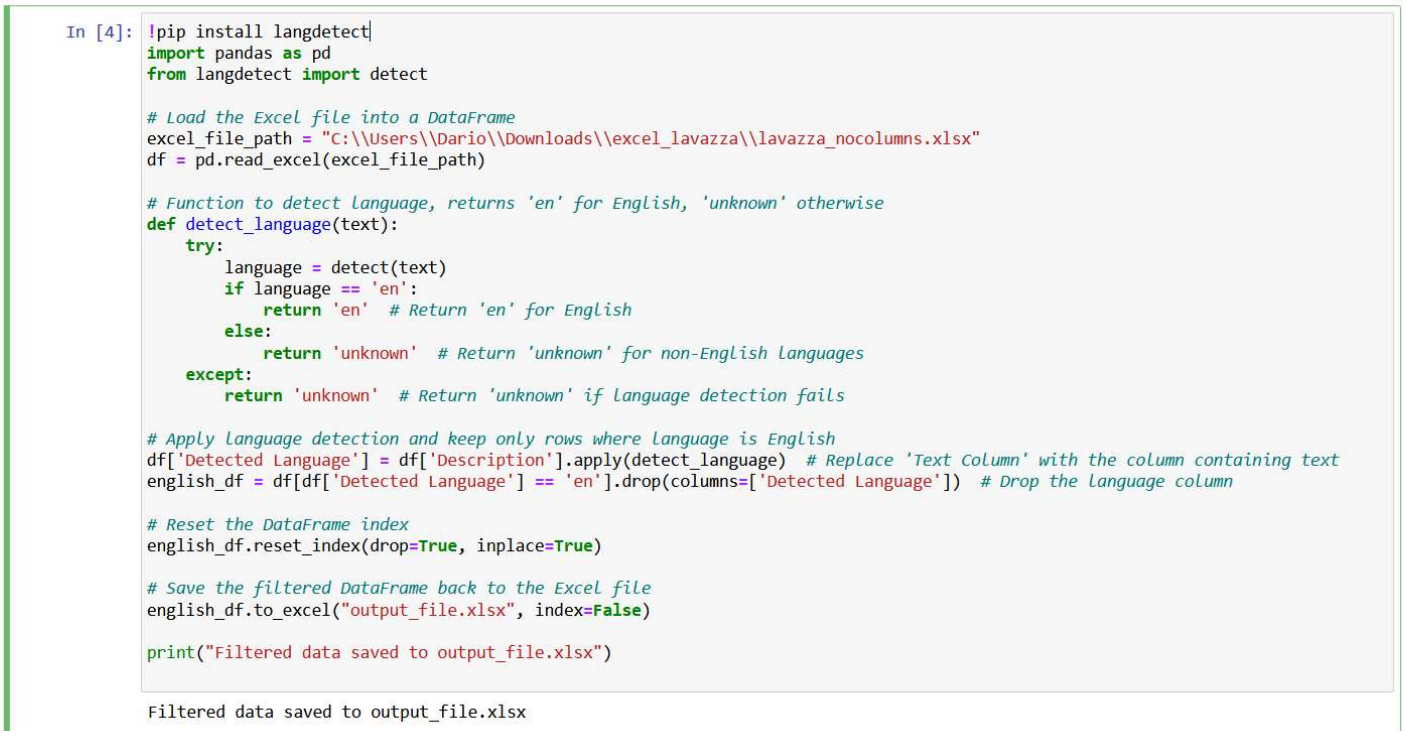
Description	Priority	Date created	Date Closed
The raw text of the ticket	A scale from 1 to 4 where 4 is low and 1 is very high	Date of issue of the ticket form the user	Date of resolution by the service desk

Table 2: Database columns

Tickets were ordered by row, each row containing the ticket information in each cell as seen in Table 2. Tickets for the whole year were collected in all languages used by Lavazza and its subsidiaries all over the world such as: Italian, English, German and French. For the scope of this analysis only English tickets were taken in consideration, all others were deleted, how will be elucidated in chapter 5.2.

5.2 Preparation of the dataset

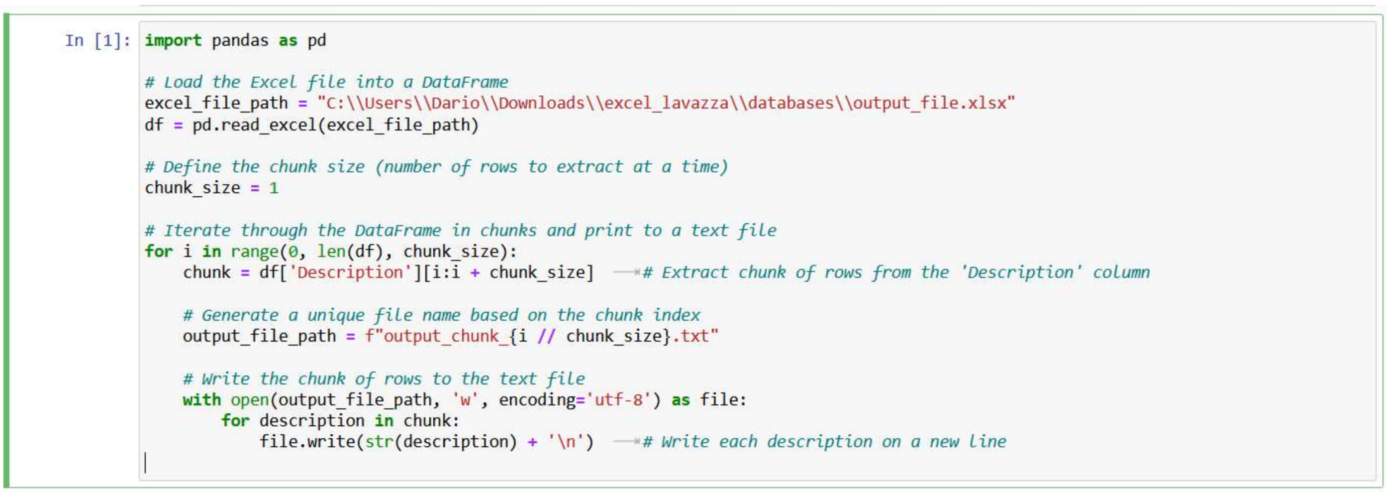
Since the only meaningful information for the topic modeling script is the raw text data contained in the "Description" column, all other columns were removed from the excel file.
Out of the 67205 tickets collected in the year 2020, all languages but English were removed, leaving 17735 rows, each containing a ticket in the English language. This process was carried out by the following python script: Figure 8.



```
In [4]: !pip install langdetect
        import pandas as pd
        from langdetect import detect

        # Load the Excel file into a DataFrame
        excel_file_path = "C:\\Users\\Dario\\Downloads\\excel_lavazza\\lavazza_nocolumns.xlsx"
        df = pd.read_excel(excel_file_path)

        # Function to detect language, returns 'en' for English, 'unknown' otherwise
        def detect_language(text):
            try:
                language = detect(text)
                if language == 'en':
                    return 'en'  # Return 'en' for English
                else:
                    return 'unknown'  # Return 'unknown' for non-English languages
            except:
                return 'unknown'  # Return 'unknown' if language detection fails

        # Apply language detection and keep only rows where language is English
        df['Detected Language'] = df['Description'].apply(detect_language)  # Replace 'Text Column' with the column containing text
        english_df = df[df['Detected Language'] == 'en'].drop(columns=['Detected Language'])  # Drop the language column

        # Reset the DataFrame index
        english_df.reset_index(drop=True, inplace=True)

        # Save the filtered DataFrame back to the Excel file
        english_df.to_excel("output_file.xlsx", index=False)

        print("Filtered data saved to output_file.xlsx")

        Filtered data saved to output_file.xlsx
```

Figure 8: langdetect python script to filter out all languages but english in an .xlsx file

Moreover, since the topic modeling software requires all tickets to be their own standalone file and to be in the .txt format, another python script (Figure 9) was used to extract row by row the "Description" column from the .xlsx file and outputting .txt files for each ticket. The output was a total of 17735 .txt documents containing the tickets written in English during the year 2020.



```
In [1]: import pandas as pd

        # Load the Excel file into a DataFrame
        excel_file_path = "C:\\Users\\Dario\\Downloads\\excel_lavazza\\databases\\output_file.xlsx"
        df = pd.read_excel(excel_file_path)

        # Define the chunk size (number of rows to extract at a time)
        chunk_size = 1

        # Iterate through the DataFrame in chunks and print to a text file
        for i in range(0, len(df), chunk_size):
            chunk = df['Description'][i:i + chunk_size]    # Extract chunk of rows from the 'Description' column

            # Generate a unique file name based on the chunk index
            output_file_path = f"output_chunk_{i // chunk_size}.txt"

            # Write the chunk of rows to the text file
            with open(output_file_path, 'w', encoding='utf-8') as file:
                for description in chunk:
                    file.write(str(description) + '\n')    # Write each description on a new line
```

Figure 9: python script to extract each row of the .xlsx in individual .txt files containing the ticket text data from the "Description" row

It is noted that encoding plays a crucial role here, if encoding settings are overviewed, loading of the dataset will fail due to the default encoding of Windows. All files, be it in .xlsx or .txt format, must be UTF-8 encoded, and so was specified in the last script (Figure 9).

Assuming the software is implemented by companies to address tickets, the script should be run at least once a day to categorize the requests. Following this reasoning the tickets from the day 02/01/2020 were chosen, for a total of 146 tickets. Out of these some outliers were removed either for being mistakenly sent or for containing meaningless information for the scope of the analysis.

5.3 Stop-word list

For better results, it is advised by the topic modeling software to have an external list containing stop words to be removed from the analysis. For the first iteration the list of most common English stop-words was taken directly from the GitHub of the software. Later, after a few iterations, a personal list was created and added to the first, this included but was not limited to: personal names, corporate specific slang and names, common URLs to company's intranet and names relative to periods of the year, adding on this, since the day of collection was January the 2nd, it was needed to remove both "January", "Jan" and "Christmas" from the list of tokens.

5.4 Parameters

To kickstart the model, the choice of topics' number (k) was made arbitrarily to initiate the modeling process, intending to create a preliminary framework for categorizing service desk tickets. While the determination of the optimal number of topics typically involves more nuanced techniques such as coherence measures or cross-validation, an initial arbitrary selection was made to initiate the exploration of latent themes within the data.

As more iterations with different k progressed, the metric most fit to evaluate the goodness of the choice was the correlation between topics, meaning how similar each self-generated topic was with the others.

Thanks to attentive reading of all the sample of tickets fed to the script, it became clear that most of them showed very similar or identical patterns, for example antivirus port scan detections (more on that in the Data Discussion chapter).

Specifically, we could identify six types of requests, so numerous tests were made with this value of k. In the end, through trial and error, it became apparent that number of topics equal to 5 was a better suit, as it had the better balance of capturing the essential themes within the dataset while ensuring non-overlapping and coherent topics, indeed topic coherence went down when using this value instead of 6, and notably each ticket was less spread between different topics.

Another value to arbitrarily choose is the number of iterations of the script. This is a fundamental step, that involves striking a good balance between computational speed and quality of the generated model.

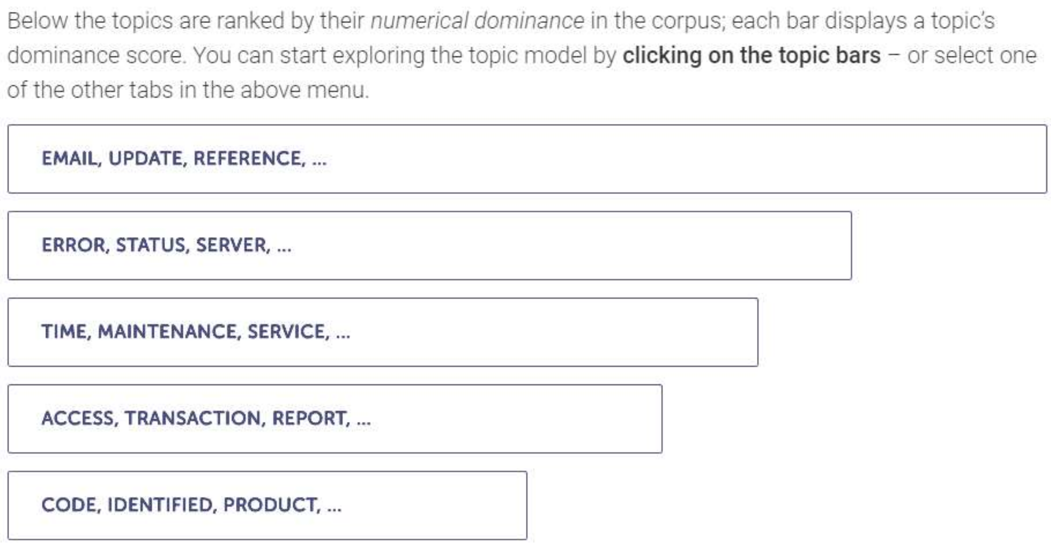
For this research, the specific value of 20,000 iterations was chosen as it struck an effective balance: firstly, the computational time required to complete 20,000 iterations was notably efficient, taking less than a minute to execute, this was helpful to allow for faster testing of the model; additionally, selecting a relatively high number of iterations ensured, usually the minimum advised is 1000, the model was able to capture the underlying patterns in the service desk ticket data accurately. It is good to note that increasing past a certain threshold, results in barely marginal improvements. This occurs because the algorithm converges to an optimal solution within a specific number of iterations, after which additional iterations do not significantly enhance the quality of the topics identified. After reaching this convergence point, the computational resources required for further iterations might outweigh the minimal gains in model accuracy.

6 Data results

The data out from Dariah topics explorer can be viewed both from the software user interface and from the output .xlsx files that will be attached to this thesis. In this chapter the outputs will be shown and, in the chapter right after, discussed.

6.1 Topics

As soon as the iterations end, the software will display at glance all topics and their relative distribution inside the corpus, from most found to least found, as shown in Figure 10.



Below the topics are ranked by their *numerical dominance* in the corpus; each bar displays a topic's dominance score. You can start exploring the topic model by **clicking on the topic bars** – or select one of the other tabs in the above menu.

EMAIL, UPDATE, REFERENCE, ...

ERROR, STATUS, SERVER, ...

TIME, MAINTENANCE, SERVICE, ...

ACCESS, TRANSACTION, REPORT, ...

CODE, IDENTIFIED, PRODUCT, ...

Figure 10: topics automatically generated by the software

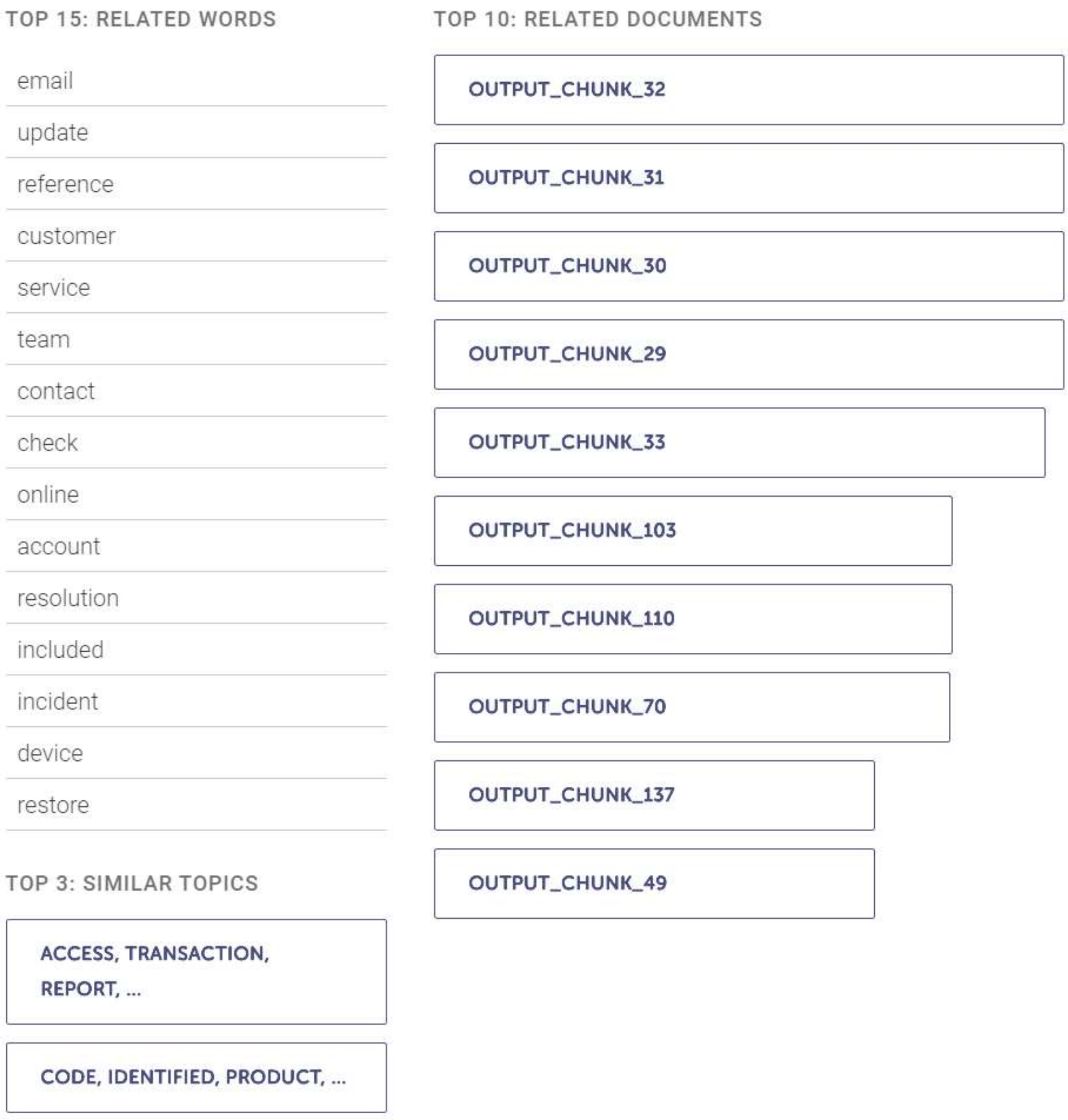
The automatically generated topics were the following, see also Figure 10:

1. Email; update; reference; customer.
2. error; status; server; code.
3. time; maintenance; service; port.
4. access; transaction; report; team.
5. Code; identified; product; uniquely.

The tokens are ordered starting from the one with the higher distribution, this could be helpful to determine at glance which categories of tickets were sent the most that day.

Note that only the four most present ones are on the list. The complete distribution of tokens can be observed in the attached excel file called topics.xlsx.

By clicking on a topic, a window (Figure 11) showing the most correlated words to the topic in question will appear. This section will also inform the user about which documents display the greatest fit and the three most correlated topics.


TOP 15: RELATED WORDS

email
update
reference
customer
service
team
contact
check
online
account
resolution
included
incident
device
restore

TOP 10: RELATED DOCUMENTS

OUTPUT_CHUNK_32
OUTPUT_CHUNK_31
OUTPUT_CHUNK_30
OUTPUT_CHUNK_29
OUTPUT_CHUNK_33
OUTPUT_CHUNK_103
OUTPUT_CHUNK_110
OUTPUT_CHUNK_70
OUTPUT_CHUNK_137
OUTPUT_CHUNK_49

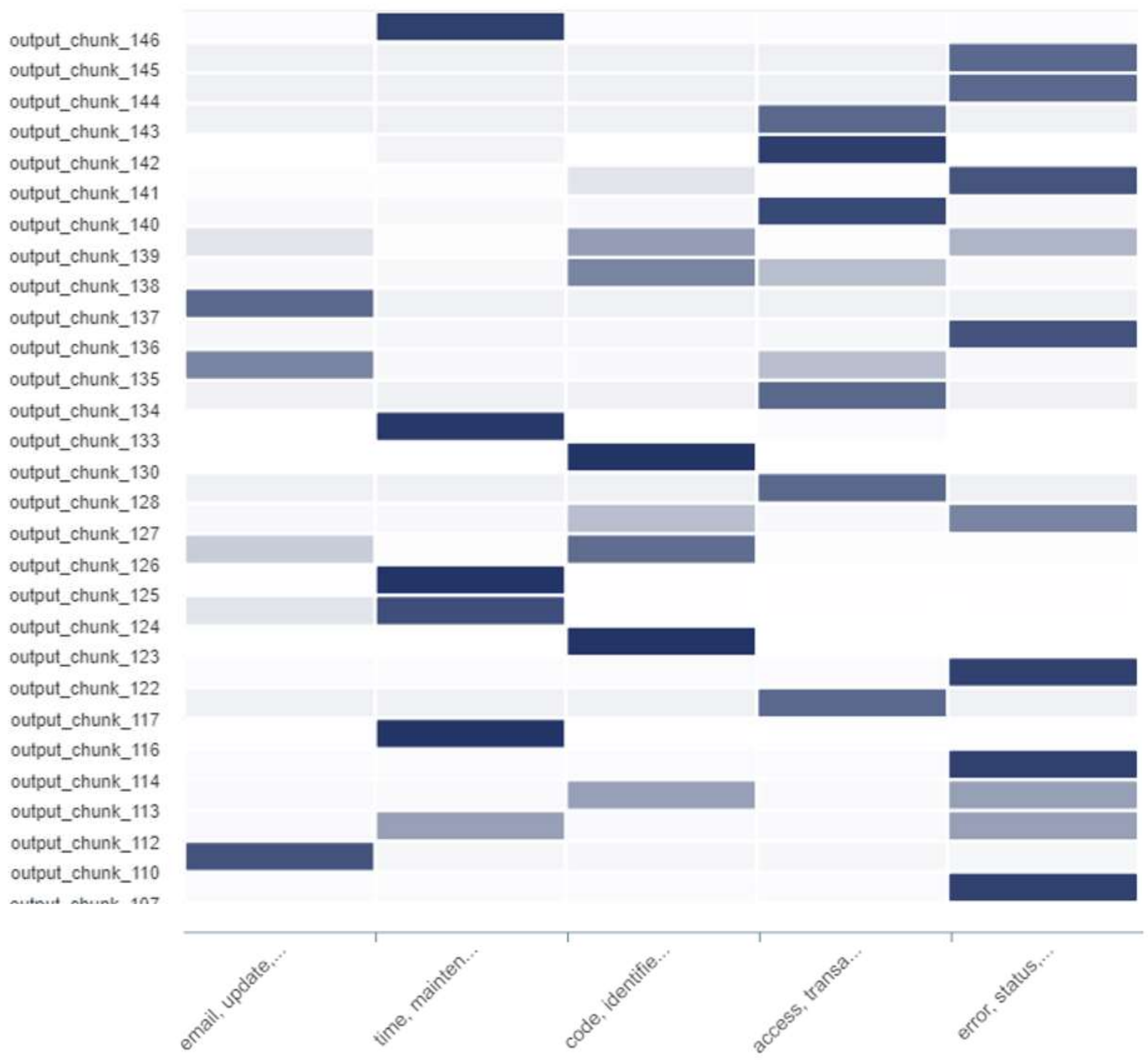
TOP 3: SIMILAR TOPICS

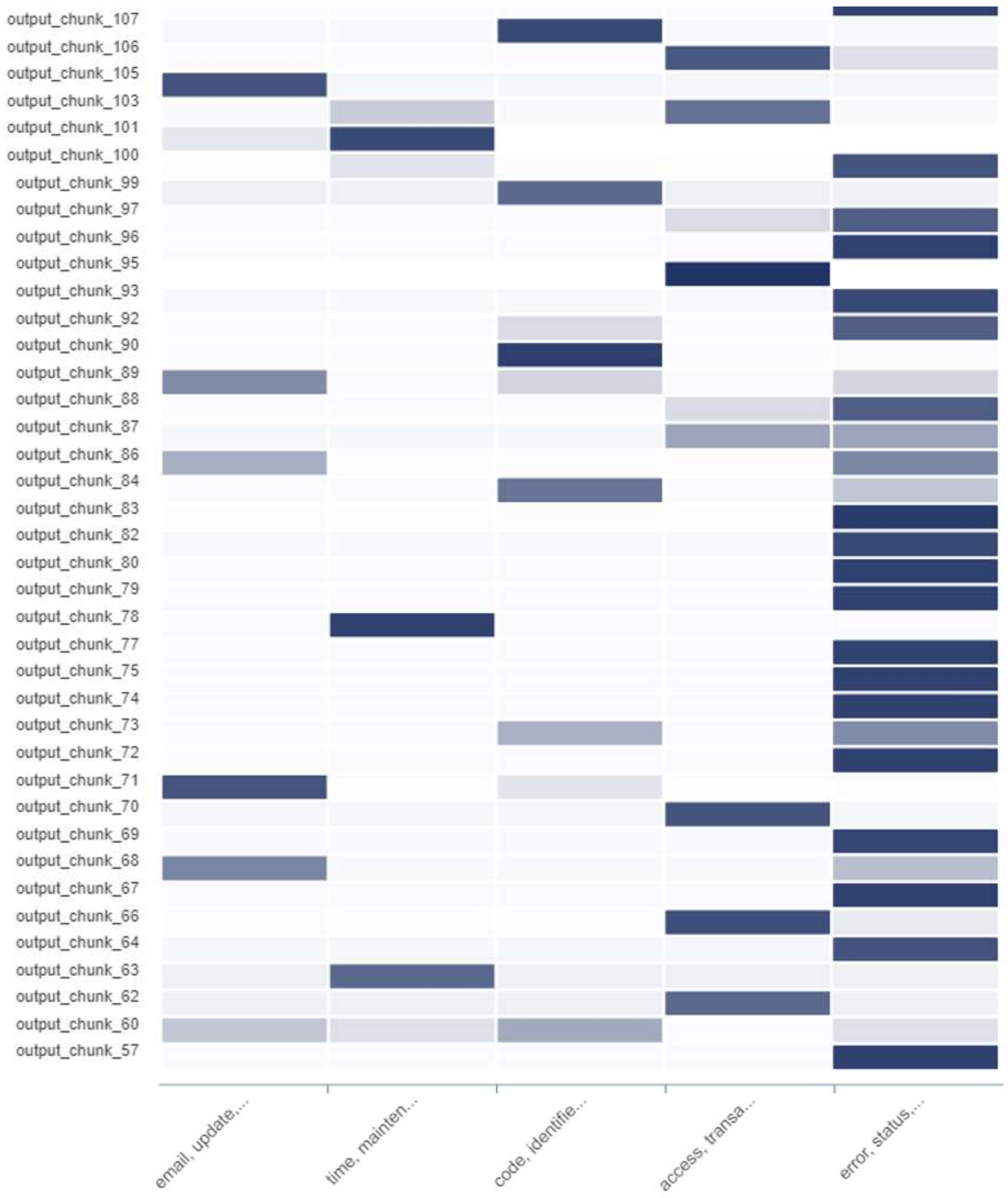
ACCESS, TRANSACTION, REPORT, ...
CODE, IDENTIFIED, PRODUCT, ...

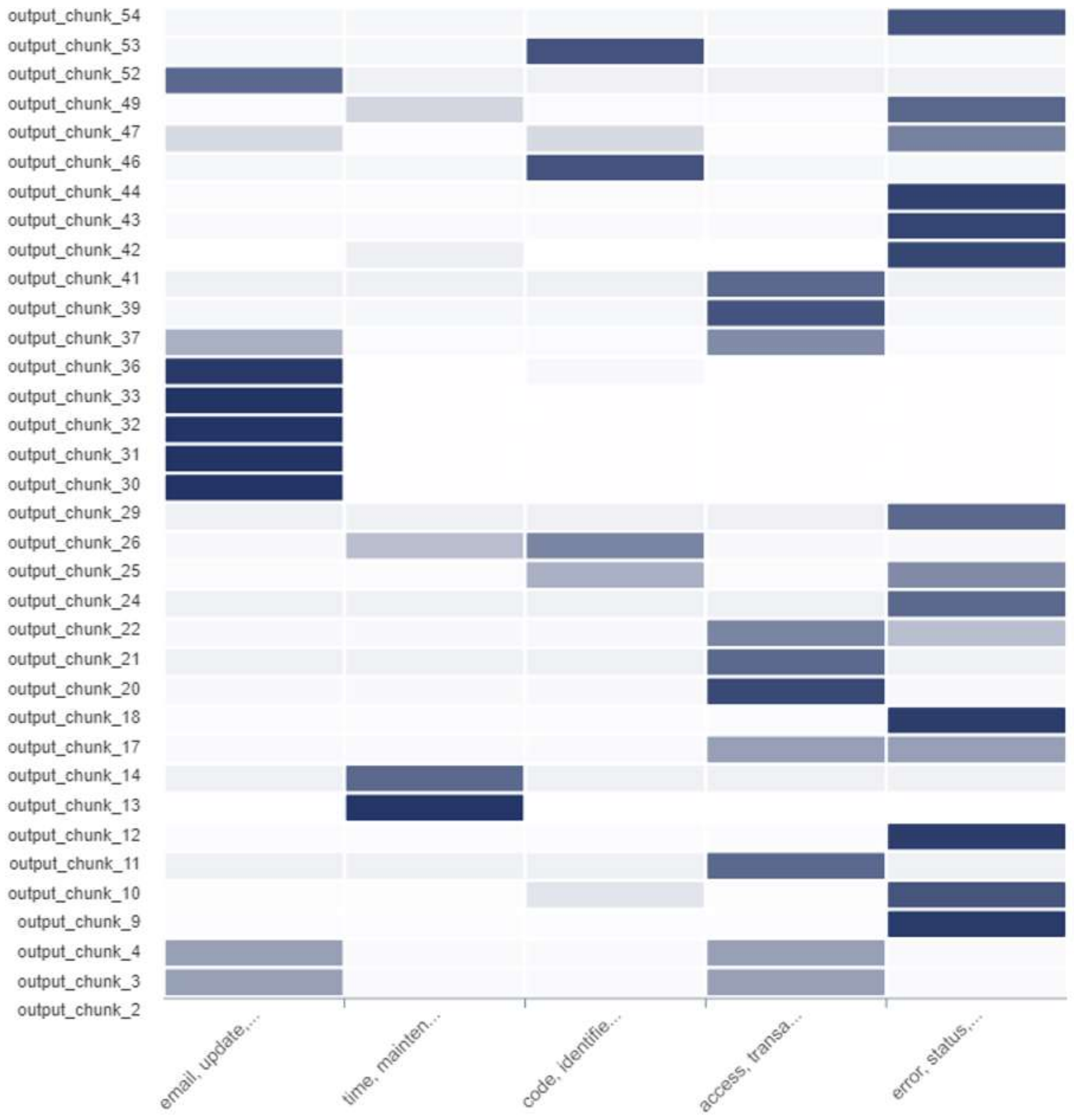
Figure 11: in depth view of a topic

6.2 Document-topic-distribution

To have a better overview of the results, the document-topic distribution matrix below, can also be displayed inside the software's interface, this graph, with each ticket on the vertical axis and topics on the horizontal one, shows thanks to the color blue (the darker the higher the correlation), how correlated each document is to which topic, this is the most important statistics, as it allows users to get to know the most relevant words detected by the script in a document without having to read it.




output_chunk_107
output_chunk_106
output_chunk_105
output_chunk_103
output_chunk_101
output_chunk_100
output_chunk_99
output_chunk_97
output_chunk_96
output_chunk_95
output_chunk_93
output_chunk_92
output_chunk_90
output_chunk_89
output_chunk_88
output_chunk_87
output_chunk_86
output_chunk_84
output_chunk_83
output_chunk_82
output_chunk_80
output_chunk_79
output_chunk_78
output_chunk_77
output_chunk_75
output_chunk_74
output_chunk_73
output_chunk_72
output_chunk_71
output_chunk_70
output_chunk_69
output_chunk_68
output_chunk_67
output_chunk_66
output_chunk_64
output_chunk_63
output_chunk_62
output_chunk_60
output_chunk_57
email, update,...
time, mainten...
code, identifie...
access, transa...
error, status,...



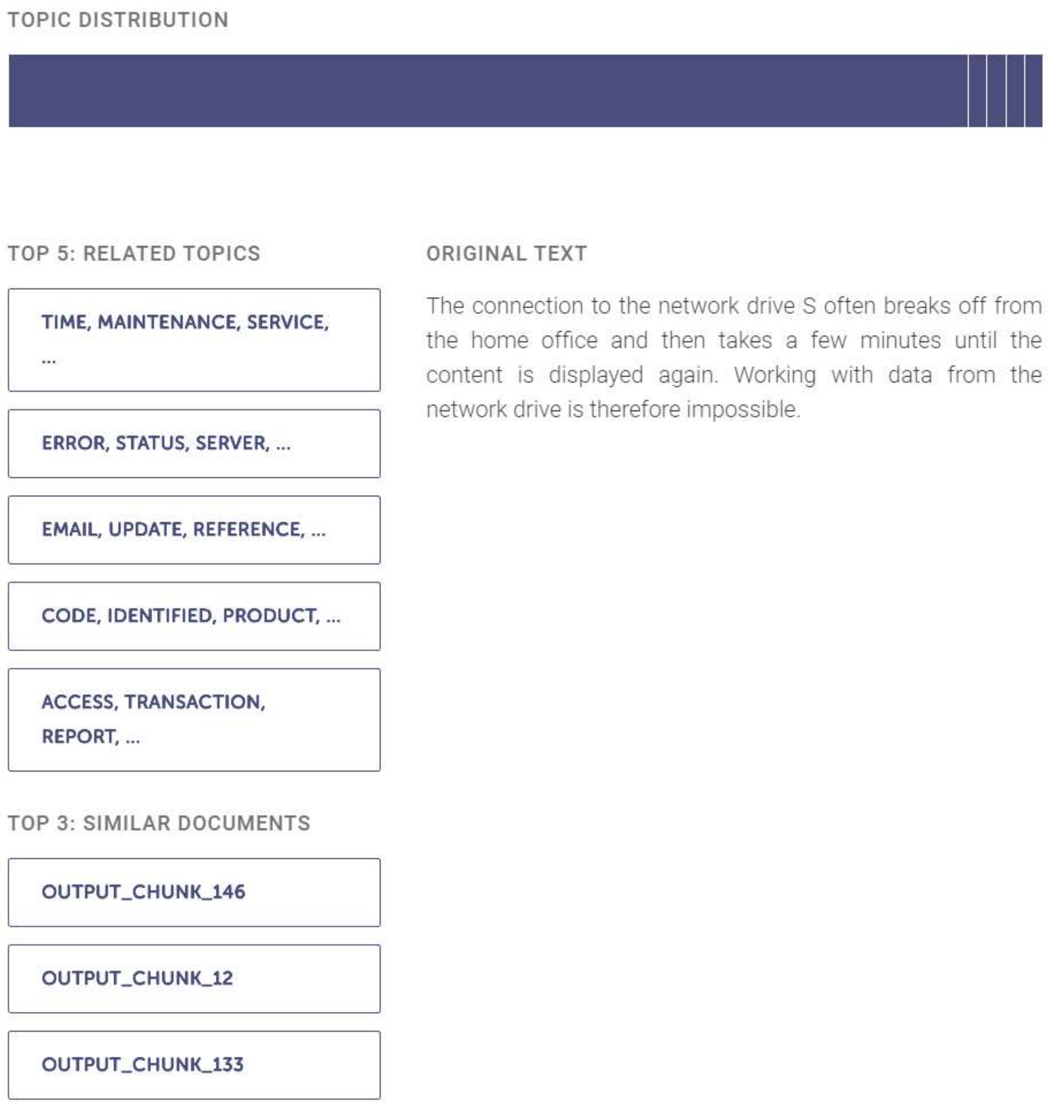
6.3 Topic similarity

This output, only found in the topic-similarities.xlsx, shows how correlated each topic is to the others, as mentioned before, the lower the inter-correlation the more non-overlapping and coherent the topics are. For example, topic 1 shows a 0.0908344783 correlation with topic 3.

6.4 Document similarity

The document-similarities.xlsx displays how similar each ticket is to the others. For example ticket 100 shows a 0.9898 correlation with respect to ticket 12, indeed the tickets are both alerts of a scheduled maintenance on the server.

While the data in the excel file is useful, it is quite cluttered and hard to navigate. Instead, right from the user interface, if users were to click on a ticket from the list of documents, its content will be displayed and with both the topic distribution and similar documents (but not the correlation value, which can only be found in the excel), as shown in Figure 12.


TOPIC DISTRIBUTION
TOP 5: RELATED TOPICS
TIME, MAINTENANCE, SERVICE, ...
ERROR, STATUS, SERVER, ...
EMAIL, UPDATE, REFERENCE, ...
CODE, IDENTIFIED, PRODUCT, ...
ACCESS, TRANSACTION, REPORT, ...
TOP 3: SIMILAR DOCUMENTS
OUTPUT_CHUNK_146
OUTPUT_CHUNK_12
OUTPUT_CHUNK_133
ORIGINAL TEXT

The connection to the network drive S often breaks off from the home office and then takes a few minutes until the content is displayed again. Working with data from the network drive is therefore impossible.

Figure 12: in depth document view

7 Data Discussion

In this chapter we will try to draw conclusions about the legitimacy of this LDA modeling as a tool for categorizing corporate tickets.

Right from the list of topics, we can see that no keywords are repeated twice. Indeed, this is a good signal of accuracy of the model and can also mean that the choice of *k,* number of topics was well taken.

Also, from the topic-similarities.xlsx, which highlights the correlation between topics, the highest value observable is 0.17314323950000002 relative to topic 5 and 3. This means that, aside from a 17% similarity between these two, which is not abnormal by any mean, all topics are significantly different from each other, another symptom of the good choice of k, and the number of iterations.

Let’s delve deeper into the topic-document distribution, to find some meaning from the most prevalent words inside each topic:

1. Topic 1: email, update, reference, customer, but also incident, account, resolution.
   The first documents strongly correlated to topic 1 (namely, number 30,29,31,32 and 33) are all automated email messages from the server giving to the user and update report on their previous ticket. They warn the user that the issue they were experiencing was due to disconnection or service outage and that the IT team was working on a resolution. The other tickets strongly related are users asking for an update from the support about unresolved problems of various nature.

2. Topic 2: error, status, server, code, but also file, message, invoice, document. Documents relative to topic 2 are from users requesting resolution either of document posting problems (ticket 4), or about server errors and sharing with the support desk the status code for an easier resolution. As we can see from the topic-distribution matrix, this topic is not that specific and correlated to just a few types of tickets. But we will go back to this point later in this chapter.

3. Topic 3: time, maintenance, service, port, scan, detected and malware. Regarding this topic, a lot of documents are an automated message from McAfee, warning the user about a port scan, malicious activities aimed at identifying weak ports on a network device, for example a laptop or a smartphone. Others again warn the user of a scheduled maintenance at a specific time of the day.

Two tickets are from users lamenting time outs in the connection to the network, this is why network is a token also present in the topic, but with a weaker probability.

4. Regarding topic 4: access, transaction, report, team, iPad, laptop, task, user. These user tickets require action from the customer support. The majority of them are from managers requiring access to the right domains and permissions for their newly entered team members. There are requests to set up iPads and laptops to the right desk, sometimes with attached requests of access to the system.
   Others were users requesting access for a specific user to be given access to corporate credit card transaction.

5. Topic 5: code, identified, product, uniquely, sap, sales, change. Some of the tickets that display strong correlation to topic 5 are automatic emails sent by SAP, because of the server experiencing difficulties with identifying specific product codes withing the monitoring or order processing system. Also stating out the respective unique product identifying codes (ZUS IDs).
   Some are from users requesting visibility in the "Supervisor Portal" of new orders and their ID codes. In general they are all relative to users presenting codes (either product, customer or company codes) to solve their issues, most of them are relative to the sales area.

From the matrix, paying attention to the coloration in the document-topic matrix, we can see that some topics are more represented than others, for example error, status, server (topic 2) is the most representative by far. Indeed, as we said before, it is not representative of just a category of tickets, but it fits well with most tickets because of the usage of very frequent words like error, server, code, status and file. This can be problematic, as we would prefer a less one-sided representation of the corpus. To solve this issue, increasing the number of topics k could be a possible solution, although other indicators may suggest otherwise.

Also visible from in the document-topic matrix, it is clear how some tickets show better results than others, for example ticket 140 shows a 99% distribution with the topic 5. Indeed, this ticket is the automated message signaling to the user that the monitor they have ordered could not be identified by its product code.

In other tickets we see an equal distribution for two different topics. For example, in number 2, we see a 0.47 distribution for both topic 4 and 1. In this ticket, a user is requesting an email update (topic 1) on the access of a new hire (topic 4). This kind of results should not worry, as the symmetrical weight for both topics means a good split between the two, and the modeling is still useful to understand the meaning of the ticket, as it will be centered on both categories.

As said before and shown in the attached “topic-similarities.xlsx”, all the topics display low correlation between them, this is encouraging, as lack of correlation between topics indicates that the LDA model has successfully identified non-overlapping and coherent themes within the corpus. Each topic is representative on its own of a specific category and there is clear separation between these.

But, as illustrated by the matrix, some topics are heavily represented, while others very little. This is also due to the fact that, together with human tickets, in the database resulted many automatic messages that clutter the corpus. It could also be considered to remove automatic tickets from the corpus, but this should be a business-related decision, to better address the big load of automated replies. Personally, I believe that such tickets should be addressed differently, regarding the server outages, for example, they should be visible and announced on the server page right from the website. For what concerns the McAfee security breaches, I believe that only the IT personnel should be notified, not the user of the device that was breached. Lastly, regarding tickets noticing the user of progress made on their requests, I believe only the solution/answer should be sent to the user, else they might tire of reading automated messages.

Some of the difficulties regarding this research were based on the elimination of outliers or too short and erroneous tickets. These, at the first iterations of the software, were giving skewed results and topics cluttered of non-inherent words, a good choice of stop words and tailoring of these after iterations was needed.

Overall, this tool shows promising results in aiding the dispatching of support tickets, though it must be noted that it can’t be a one-time solution, as it requires constant tailoring to be of use and improve the topics further.

I believe, since the huge daily influx of tickets, that a better solution would be to start iterations with a larger database, maybe comprising the tickets of the whole past week, to create a baseline based on a large dataset of past requests.

After this baseline work and preprocessing (for example identifying and keeping updated a database of specific stop words, or trying different topic numbers *k*), this tool would be best used daily, run at the start of the working day to give the support desk a better insight of their daily work ahead.

8 Conclusion

The objective of this thesis was to explore the potential use of topic modeling algorithms, specifically applying Latent Dirichlet Allocation (LDA), in the realm of service ticket categorization in large corporate environments. The goal was set the baseline for adoption of a technology able to enhance the efficiency and speed of ticket resolution, by relying on an unsupervised algorithm.

The research implemented LDA using “Dariah-DE Topics Explorer”, a software implementing topic modeling with a user-friendly interface and without the need of coding knowledge from the user. The model was trained on a large dataset of service desk tickets, making it able it to automatically identify a broad range of topics that can significantly aid in automatic ticket categorization.

The findings may pave the way for a new staple in IT and have important results in improving efficiency in service desk operations. By automating ticket categorization, firms can ensure faster resolution of queries, bringing an increase in productivity and cost efficiency. Software implementations like such could potentially revolutionize the way service desks operate.

However, this study does not come without limitations. The effectiveness of the LDA-based system may vary depending on the quality and diversity of the dataset used for training, for example short and meaningless tickets, while also long and system generated ones can clutter the database. For this reason, the system would require continuous optimization, stricter preprocessing, elimination of all outliers and tailoring both the list of stop words and fixed parameters (k number of topics and number of iterations) to every diverse company.

It is clear that, provided larger budgets and resources in possession of firms, the better way to implement LDA for the above-mentioned operations, would be best done writing the script from scratch using Python or R, these methods would provide better customization and better accuracy, while also not being limited to just the output given by the all-in-one software used for this dissertation.

Nonetheless, from the experiments carried out during this work, some perplexities have arisen regarding the way. Especially regarding how automated tickets were handled, directly sent to the users and back to the service desk server to be kept for collection. But, during the workday, too many tickets were automated, with the risk of users failing to check them and be notified. Also, these tickets were still found in the database at the end of the day, making it harder for topic modeling algorithms to function properly.

Personally, I believe a better way to organize service desk queries would be to eliminate this first level categorization set by the user, as users can always make mistakes due to inexperience in the field they are seeking help for. A better method would be to keep all tickets as free text and implementing a topic modeling software to do the categorization on its own, reducing the workload on both users and the service desk operators. Automatic tickets could be easily sorted by free text technologies already available and present in our everyday life, for example in spam detection in emailing platforms.

Hopefully, future research could explore the integration of the LDA-based system within already existing or completely reinvented software for service desk operations. This would improve the usefulness of these software by complementing their already established methods of prioritization, ticket data collection and statistics with an automated and unsupervised assignment of topics or better yet categories to each inbound query.

In conclusion, while on the one hand this study accomplishes its goal of shedding more light into the employment of topic modeling algorithms for enhancing the dispatching service desk tickets. On the other hand, it is important to acknowledge the limitations, such as the quality of the database, the intrinsic variability and specificity proper of every business and the need of proper baselining of the algorithms and later constant optimization and improvement to gain always better and coherent categorization.

9 References

[1] Feras Al-Hawari, Hala Barham, “A machine learning based help desk system for IT service management”, *Journal of King Saud University - Computer and Information Sciences*, Volume 33, Issue 6, 2021,Pages 702-718,

[2] W. Zhao, J. J. Chen, R. Perkins, Z. Liu, W. Ge, Y. Ding, and W. Zou, “A heuristic approach to determine an appropriate number of topics in topic mod- eling,” in *BMC bioinformatics*, vol. 16, no. 13. Springer, 2015, p. S8.

[3] N. Mukherjee, S. Neogy, and S. Chattopadhyay, *Big Data in Ehealthcare: Challenges and Perspectives*. CRC Press, 2019. [Online]. Available: https://books.google.no/books?id=o 6PDwAAQBAJ

[4] A. Hotho, A. Nu¨rnberger, and G. Paaß, “A brief survey of text mining.” In *Ldv Forum*, vol. 20, no. 1. Citeseer, 2005, pp. 19–62.

[5] Prof. David Blei - *Probabilistic Topic Models and User Behavior*. U. of Edinburgh (2017, Feb). Youtube. [Online]. Available: https://www.youtube.com/watch?v=FkckgwMHP2s

[6] C.D. Manning and H. Schu¨tze, *Foundations of Statistical Natural Language Processing*. MIT press, 1999.

[7] D. Kr, *Journey through Nlp Research - Basics*, Jul 2017. [Online]. Available: https://medium.com/@deepukr85/journey-through-nlp-research-part-1-2e93fdeaad9c

[8] O. Davydova, *10 Applications of Artificial Neural Networks in Natural Language Processing*, Aug 2017. [Online]. Available: https://medium.com/@datamonsters/artificial-neural-networks-in-natural-language-processing-bcf62aa9151a

[9] Sciforce, *Ai hardware and the battle for more computational power*, Nov 2019. [Online]. Available: https://medium.com/sciforce/ ai-hardware-and-the-battle-for-more-computational-power-3272045160a6

[10] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “Indexing by latent semantic analysis,” *Journal of the American society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[11] D. M. Blei, A.Y. Ng, and M.I. Jordan, “Latent dirichlet allocation,” *Journal of Machine Learning Research*, vol. 3, no. Jan, pp. 993–1022, 2003.

[12] A. Amado, P. Cortez, P. Rita, and S. Moro, “Research trends on big data in marketing: A text mining and topic modeling

based literature analysis," *European Research on Management and Business Economics*, vol. 24, no. 1, pp. 1–7, 2018.

[13] E. Niiler, *An Ai Epidemiologist sent the first Alerts of the Coro- navirus*, Jan 2020. [Online]. Available: https://www.wired.com/story/ ai-epidemiologist-wuhan-public-health-warnings/

[14] C. Stieg, *How this Canadian start-up spotted coronavirus before everyone else knew about it*, Mar 2020. [Online]. Available: https://www.cnbc.com/2020/03/03/bluedot-used-artificial-intelligence-to-predict-coronavirus-spread.html

[15] N. Ko, B. Jeong, S. Choi, and J. Yoon, "Identifying product opportunities us- ing social media mining: application of topic modeling and chance discovery theory," *IEEE Access*, vol. 6, pp. 1680–1693, 2017.

[16] H. Pillai, *Covid-19 and new age technology*, May 2020. [Online]. Available: https://www.thedispatch.in/covid-19-and-new-age-technology/

[17] A. McCallum, X. Wang, and N. Mohanty, "Joint group and topic discovery from relations and text," in *ICML Workshop on Statistical Network Analysis*. Springer, 2006, pp. 28–44.

[18] J. Chang, J. Boyd-Graber, and D. M. Blei, "Connections between the lines: augmenting social networks with text," in *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data mining*, 2009, pp. 169–178.

[19] L. Hong and B.D. Davison, "Empirical study of topic modeling in twitter," in *Proceedings of the first workshop on social media analytics*, 2010, pp. 80–88.

[20] S. Raschka and V. Mirjalili, *Python Machine Learning: Machine Learning and Deep Learning with Python, scikit-learn, and TensorFlow 2, 3rd Edition*, Birmingham – Mumbai 2019.

[21] J. Tang, Z. Meng, X. Nguyen, Q. Mei, and M. Zhang, "Understanding the limiting factors of topic modeling via posterior contraction analysis," in *International Conference on Machine Learning*, 2014, pp. 190–198.

[22] T. Beysolow II, "Topic modeling and word embeddings", in *Applied Natural Language Processing with Python*. Springer, 2018, pp. 77–119.

[23] S. Vijayarani, R. Janani *et al.*, "Text mining: open source tokenization tools- an analysis," *Advanced Computational Intelligence: An International Journal (ACII)*, vol. 3, no. 1, pp. 37–

47, 2016.

[24] V. Balakrishnan and E. Lloyd-Yemoh, *Stemming and Lemmatization: a Comparison of Retrieval Performances*, 2014.

[25] M. Pitchford, *Preparing Your Data for Topic Modeling,* Nov 2017.[Online].Available:https://publish.illinois.edu/commonsknowledge/2017/11/16/preparing-your-data-for-topic-modeling/

[26] B. Bengfort, R. Bilbro, and T. Ojeda, *Applied Text Analysis with Python: Enabling Language-aware Data Products with Machine Learning*. " O' Reilly Media Inc., 2018.

[27] T. Mikolov, K. Chen, G. Corrado, and J. Dean, *Efficient Estimation of Word Representations in Vector Space*, arXiv preprint arXiv:1301.3781, 2013.

[28] T. Hofmann, "Probabilistic latent semantic indexing," in *Proceedings of th22nd annual international ACM SIGIR conference on Research and development in information retrieval*, 1999, pp. 50–57.

[29] D. Ramage, D. Hall, R. Nallapati, and C. D. Manning, "Labeled lda: A supervised topic model for credit attribution in multi-labeled corpora," in *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 1.* Association for Computational Linguistics, 2009, pp. 248–256.

[30] J. D. Mcauliffe and D. M. Blei, "Supervised topic models," in *Advances in Neural Information Processing Systems*, 2008, pp. 121–128.

[31] J. Kim, M. Park, H. Kim, S. Cho, and P. Kang, "Insider threat detection based on user behavior modeling and anomaly detection algorithms," *Applied Sciences*, vol. 9, no. 19, p. 4018, 2019.