Article

Not peer-reviewed version

Enhancing Python-based Sentiment Analysis: Empowering Industrial Engineers and Managers in the Service Industry

João Reis *

Posted Date: 15 August 2024

doi: 10.20944/preprints202408.1131.v1

Keywords: Python; sentiment analysis; requests library; textblob; machine learning; web-scraping libraries; lexicon-based approach; AI applications.

Article

Enhancing Python-Based Sentiment Analysis: Empowering Industrial Engineers and Managers in the Service Industry

João Reis [1,2,*]

[1] Industrial Engineering and Management, Faculty of Engineering & RCM2+, Lusófona University, Campo Grande, 1749-024, Lisbon, Portugal
[2] Research Unit on Governance, Competitiveness and Public Policies (GOVCOPP), Aveiro University, Campus Universitario de Santiago, 3810-193, Aveiro, Portugal
* Correspondence: joao.reis@ulusofona.pt

Abstract: Python, a versatile programming language, holds vast potential for Sentiment Analysis (SA). Leveraging the Requests and TextBlob libraries, we have crafted a user-friendly code that enables Industrial Engineers (IE) and managers, particularly those new to Python, to extract and analyze sentiment efficiently. We aim to provide IE/managers in service companies requiring SA capabilities with a simple yet effective Python solution. While machine learning resources like PyTorch/TensorFlow are commonly utilized in SA, offering pre-built algorithms and tools for training, and implementing machine learning models, we sought to exploit Python's versatility by integrating additional web-scraping libraries. Thus, using a lexicon-based approach, we intend to deliver an informative and practical article. The code in this article is twofold; firstly, it can be easily adapted by IE/managers possessing basic Python skills; secondly, we aim to inspire junior IE/managers to develop their own customized coding solutions tailored to specific organizational needs.

Keywords: python; sentiment analysis; requests library; textblob; machine learning; web-scraping libraries; lexicon-based approach; AI applications

1. Introduction

The temporal gap between industrial revolutions has been progressively decreasing over time, indicating that universities will need to adjust and evolve their industrial engineering curricula at a faster pace to meet organizational demands. Consequently, universities that thrive in the future will likely be those that boldly introduce innovative curricula tailored to the needs of tomorrow. This endeavor requires courage to design curricula that explore emerging areas of knowledge. A possible solution to this challenge could involve bridging the gap between service sector organizational requirements, curricula offerings, and students' aspirations for future employability [1]. Thus, it is necessary to implement a curricular reform in industrial engineering courses, moving from a syllabus focused on manufacturing to a more balanced model oriented toward services – this statement builds on the argument that modern economies are service-based, rapidly growing, and increasingly complex [2]. This shift to a service-oriented approach is primarily driven by digital transformation [3,4] and the adoption of innovative technologies like Artificial Intelligence (AI) [5,6]. Typically, industrial engineering curricula in universities primarily emphasize theoretical and academic knowledge, thereby neglecting the importance of practical application. Consequently, industrial engineering students often lack the skills required by service companies, forcing them to rely on on-the-job training (Type-1 workers) or self-study (Type-2 workers) [7]. Moreover, the rapid advancements in technology make it challenging for universities to maintain up-to-date curricula [8,9], further exacerbating the shortage of technical skills relevant to the real job market. Lastly, the lack of collaboration between universities and businesses deprives students of valuable insights needed to understand and adapt to future work challenges effectively. Considering the rapidly

 doi:10.20944/preprints202408.1131.v1

evolving technological landscape, the emergence of new high-level and general-purpose programming languages has greatly facilitated Industrial Engineers (IE) and managers in delivering top-notch services.

In today's digital era, service companies and state services have recognized the potential of leveraging customer feedback and employing advanced technologies to enhance their offerings [10]. As businesses increasingly rely on digital resources [11], they have come to realize the value of harnessing customer opinions to drive service improvements. By utilizing cutting-edge technologies, these companies can effectively capitalize on customer insights and leverage them to optimize their services. This is where SA or Opinion Mining (OM) gains relevance, as it aims to collect, monitor, and analyze the sentiments [12], opinions, attitudes, and emotions expressed by individuals toward service delivery [13]. Therefore, SA offers valuable insights into customer opinions [14], preferences [15], and experiences [16].

There are several compelling reasons for utilizing SA in the service industry. Firstly, it enables companies to measure customer satisfaction effectively [17], aiding in the assessment of overall satisfaction levels. By comprehending the user-generated data, organizations can pinpoint areas that require improvement, leading to enhanced customer satisfaction [17]. Secondly, SA assists in managing omnichannel presence and brand image [18]. By monitoring sentiment across various digital platforms, companies can proactively handle their brand perception [19]. This enables them to leverage marketing strategies to highlight the positive aspects of their brand and maintain a favorable image. Thirdly, SA provides an opportunity to gain a competitive edge in the market. By analyzing the sentiment of customers toward competitors, service companies can identify strengths and weaknesses and adapt their strategies accordingly [12]. By monitoring sentiment on social networks and other platforms [20], companies can operate within proactive systems, promptly identifying issues and defining appropriate actions. This minimizes customer dissatisfaction and safeguards the company's reputation. Overall, SA serves as a valuable tool for empowering service sector companies to meet customer needs effectively.

Python offers numerous opportunities for SA, thanks to its extensive ecosystem of libraries and tools designed for Natural Language Processing (NLP) [21]. Well-known libraries such as NLTK (Natural Language Toolkit) [22] and TextBlob provide pre-built functions and templates for text processing and SA, saving development time. We choose Python for SA because it prioritizes simplicity and ease of understanding, making it accessible for junior IE and managers. Furthermore, Python benefits from an active developer community that freely shares tutorials and comprehensive documentation, enabling the readers of this article to delve deeper into the topic. Lastly, Python facilitates integration with other technologies and can be used alongside popular frameworks like Flask or Django to develop web applications or SA APIs, ensuring continuity in this article.

To bridge the gap between service sector organizational needs and the career aspirations of industrial engineering students in SA we draw the following research question – How to develop sentiment analysis code using Python?

This paper is organized as follows: Section 2 proceeds with a literature review that elucidates the functioning of SA and delves into the extent to which it can be subjective. The literature review serves to establish a foundation for the subsequent discussion. Section 3 outlines the materials and methods employed in our research, providing a comprehensive description of the research design, data sources, and analytical techniques employed to conduct the SA. Section 4 presents the results section where Python code is utilized for SA. We provide a detailed explanation of the code's implementation and discuss the outcomes derived from its execution. This section aims to showcase the practical application of SA and highlight the key findings obtained from our study. Lastly, in Section 5 we present a conclusion that encapsulates the theoretical and managerial contributions of the article. We also address any limitations encountered during the research process and offer suggestions for future research endeavors. This final section aims to provide a comprehensive understanding of the significance of our findings and their potential implications for both academia and industry.

2. Literature Review

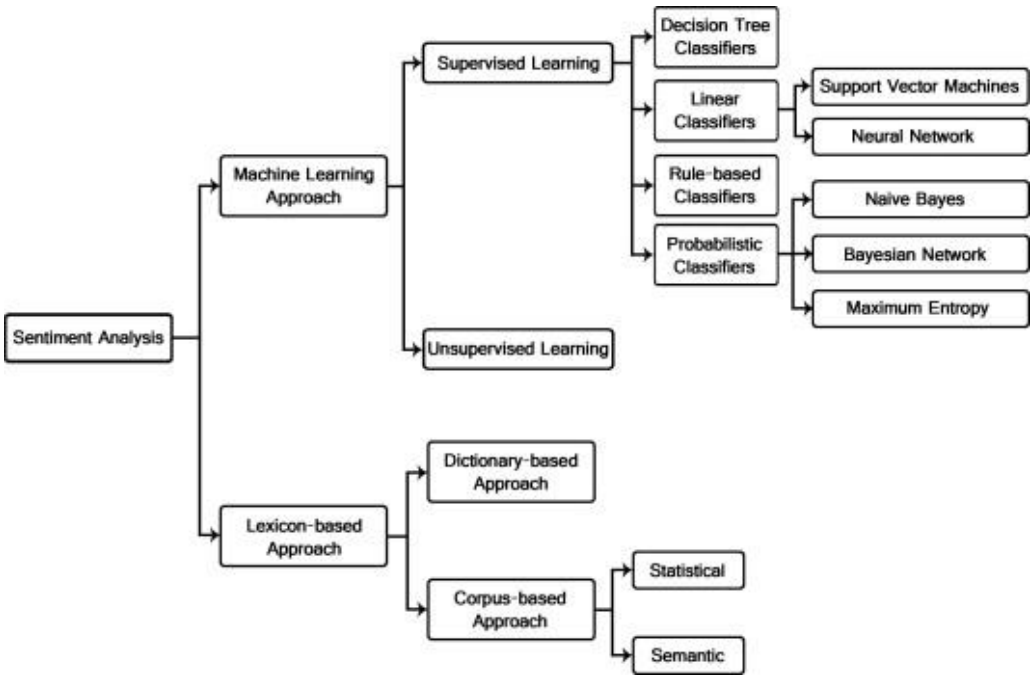
 doi:10.20944/preprints202408.1131.v1

2.1. Overview of Sentiment Analysis and Its Subjectivity

As argued, SA is a valuable tool for assessing the emotional tone conveyed in various types of text, such as social media posts and complaint portals, its main objective is to determine whether the sentiment expressed is positive, negative, or neutral. SA can be divided into three levels: document, sentence, and aspect level, depending on the degree of text granularity [23]. According to Behdenna et al. [24], document-level SA involves determining the overall opinion expressed in the whole document, as expressing a positive or negative opinion or sentiment; sentence-level SA, on the other hand, aims to ascertain whether individual sentences convey opinions. At this level, a clear distinction is made between objective sentences, which provide information, and subjective sentences, which express positive or negative views; finally, at the aspect level, a more intricate analysis is conducted, necessitating the utilization of NLP. In this case, the treatment consists of two steps: firstly, identifying the entity and its corresponding aspects, and secondly, evaluating the opinion expressed for each aspect. Medhat et al. [25] offers a comprehensive demonstration of aspect-level SA, highlighting the fact that individuals can express diverse opinions regarding various aspects of a single entity such as in the sentence "The voice quality of this phone is not good, but the battery life is long." In line with the objective of our article, which aims to prepare up-and-coming IE or managers with SA tools, we will primarily concentrate on the first category of SA.

In practical terms, SA utilizes various datasets to enhance decision-making across multiple domains. For instance, in politics [26,27], SA plays a crucial role in political campaigns and governance by evaluating public opinion; in financial services [28], SA is employed to analyze news articles and social media discussions, aiding in informed decision-making; and, in healthcare [29,30], to assess patient feedback, identify potential issues, and enhance healthcare services. These examples merely scratch the surface of SA's wide-ranging applications.

The SA initial step involves text pre-processing [31], which entails removing irrelevant information like numbers and special characters. Following, the text is tokenized, meaning that the text is divided into individual words or phrases. SA typically utilizes either a lexicon-based approach [32] (Figure 1), where a set of words or phrases is assigned positive, negative, or neutral scores, or a machine learning-based approach [33] (Figure 1). In the latter case, the pre-processed text is transformed into a suitable format for analysis by extracting relevant features. This may involve a training phase, during which a labeled dataset containing texts with their corresponding sentiment values is used to teach the model to recognize patterns and relationships between features and sentiment labels. Post-processing involves refining the results through e.g., sentiment aggregation [34] and sentiment normalization [35].



 doi:10.20944/preprints202408.1131.v1

Figure 1. Sentiment classification techniques [25].

Despite its advantages, SA remains a complex task that does not always accurately capture the sentiment expressed in a text [36]. Factors like irony and sarcasm can pose challenges in SA [13]. For instance, Kamath et al. [37] highlighted sarcasm as a prevalent form of opinion expression on social media sites and product review platforms like Amazon and Flipkart. Sarcasm involves a deliberate contrast between the intended and implied meanings, making it challenging to identify the underlying sentiment. The detection of sarcasm is crucial for NLP applications such as SA. Numerous advanced techniques and scientific articles have emerged to address this challenge, contributing significantly to advancements in this field.

Subjectivity is another crucial component of SA [38], to capture human language's intricacies and quirks. Analyzing subjectivity enables SA algorithms to comprehend the intensity of emotions conveyed in text [39] and provide sentiment interpretation. For instance, consider the following sentences:

Sentence 1. "Today it is raining".
Sentence 2. "I love the rain outside".

In Sentence 1, the statement regarding the weather is objective and based on information, devoid of opinion or emotion. In Sentence 2, positive sentiments and subjective opinions regarding the rain can be discerned. Consequently, subjectivity analysis in SA involves evaluating the degree of subjectivity in each sentence and assigning an appropriate sentiment score. While quantifying subjective opinions and emotional expressions may pose challenges, a suitable classification methodology facilitates accurate SA of textual data. To perform SA and quantify subjective opinions, several general steps can be followed: (1) define the sentiment scale by establishing a quantification scale for opinions, such as a 3-point or 4-point scale representing strongly negative to strongly positive sentiments [40]; (2) acquire labeled data, by collecting text samples that are annotated with corresponding sentiment scores. This labeled data serves as the training set for developing SA models [41]; (3) text data preprocessing, by conducting preprocessing on the text data, including the removal of irrelevant words, punctuation, and other unnecessary information [42]; (4) feature extraction, by converting the preprocessed text data into a numerical representation that can be understood by machine learning algorithms [43,44]; (5) build a SA model, by employ supervised machine learning algorithms, such as Recurrent Neural Networks (RNNs) [45,46], to train a SA model using the labeled data; (6) model training and evaluation, by splitting the labeled data into training and test sets. Train the SA model using the training set and evaluate its performance on the test set [47], utilizing appropriate evaluation metrics; (7) model application, once the SA model is trained, it can be applied to new, unlabeled text data to predict sentiment scores for subjective opinions and emotional expressions [48]. The model assigns sentiment scores based on the predefined sentiment scale; (8) fine-tuning and iterations [49], as SA models often require fine-tuning and iterative processes to improve their performance. This involves adjusting model parameters, exploring different preprocessing techniques, or considering alternative model architectures.

One important issue to consider is that while SA can provide useful insights, it may not accurately capture the entirety of subjective opinions and emotional expressions. NLP techniques have limitations, and contextual understanding and cultural nuances can impact SA results. Therefore, it is crucial to interpret the output of SA models with caution and regard them as tools that provide estimations rather than definitive measurements of subjective opinions and emotions.

2.2. Sentiment Analysis with Machine Learning- or Lexicon-Based Approach

As mentioned above in Figure 1, SA encompasses two distinct approaches [50]. One of these approaches is the machine learning-based approach, where SA is framed as a classification problem. In this approach, a machine learning model is trained using a labeled dataset. The model learns patterns and relationships between different textual resources such as words, phrases, and syntactic

 doi:10.20944/preprints202408.1131.v1

structures, and their corresponding sentiment labels during the training phase. Once the model is trained, it can accurately predict the sentiment of new, previously unseen texts. The machine learning-based approach offers several advantages. Firstly, it demonstrates adaptability, as machine learning models can be adjusted to different domains or languages by training them on relevant data. This allows the models to capture nuanced sentiment patterns specific to the target domain. Secondly, these models exhibit a higher level of textual comprehension. By considering the context in which words or phrases appear, machine learning models can provide more nuanced and subtle SA. This helps in distinguishing between similar phrases with different sentiments. Lastly, machine learning models are capable of handling new or uncommon words. Through the generalization and inference capabilities developed during training, these models can infer sentiment for words they have not encountered before, leveraging their learned knowledge of the language. Overall, the machine learning-based approach in SA offers adaptability, contextual comprehension [51], and the ability to handle new or unusual words, making it a powerful technique for accurately predicting sentiment in diverse texts.

However, the machine learning-based approach is not without its disadvantages. One significant drawback is the substantial requirement for labeled training data, which is normally supervised [52]. Obtaining a sufficiently large and accurately labeled dataset can be a time-consuming and expensive endeavor [53], particularly when accounting for cultural variations that exist across different languages and regions. This data scarcity issue poses a challenge in training robust models that can generalize well across diverse contexts. Moreover, certain machine learning models, particularly Deep Neural Networks (DNN), can be perceived as "black boxes" due to their lack of interpretability. The inner workings and decision-making processes of these models may not be easily understandable or explainable. This lack of interpretability raises concerns regarding the transparency of the SA process, making it difficult to discern the factors contributing to a specific sentiment prediction. As a result, trust and confidence in the model's outcomes may be diminished, particularly in applications where interpretability and explainability are crucial. While the machine learning-based approach offers advantages in SA, challenges related to the availability of labeled training data and the interpretability of complex models must be acknowledged. Ongoing research and development efforts are necessary to address these limitations and enhance the overall effectiveness, generalizability, and transparency of machine learning-based SA models.

The second option depicted in Figure 1 corresponds to the lexicon-based approach, which is more relevant to this article. In this approach, SA relies on pre-built sentiment lexicons or dictionaries [32]. These lexicons consist of a comprehensive collection of punctuated and labeled words or phrases, each associated with a sentiment, typically positive or negative. The sentiment score is assigned based on the semantic orientation of the word or its association with positive or negative sentiment [54]. In the lexicon-based approach, the text is analyzed word by word, and sentiment scores are calculated by aggregating the sentiment values of individual words. The overall sentiment of the text is determined based on these aggregated scores, forming the foundation for the development of our SA code. The lexicon-based approach offers several advantages [55]. Firstly, it is based on lexicons, which are relatively simple to interpret and do not require extensive training on labeled data. Additionally, lexicons provide higher interpretability compared to machine learning models, as it is easier to understand why a particular sentiment was assigned to a word or phrase based on the lexicon's predefined sentiment labels. However, the lexicon-based approach also has its limitations. It exhibits a limited understanding of context, treating each word or phrase as an independent entity. Consequently, this approach may yield less accurate SA since it does not fully consider the contextual relationships between words. Furthermore, as mentioned before, dealing with sarcasm or irony becomes challenging, as the lexicon-based approach requires a deeper understanding of the context and tone of the text to accurately capture such nuances. The effectiveness of the lexicon-based approach heavily relies on the quality and coverage of the sentiment lexicons, which may not encompass all the words or phrases relevant to SA, resulting in somewhat limited accuracy. Lastly, this approach encounters difficulties in handling sentiment in specialized domains that utilize specific vocabulary or jargon. Despite these drawbacks, we have

 doi:10.20944/preprints202408.1131.v1

chosen to follow the lexicon-based approach for our article, primarily due to its simplicity, making it well-suited for junior IE and managers. To summarize this section, the machine learning approach to SA uses data and labeled training models to learn sentiment patterns, whereas the lexicon-based approach relies on pre-built sentiment lexicons offering more robust performance across domains and text [56]. The machine learning approach offers greater adaptability and contextual understanding; however, it also requires labeled data. In contrast, the lexicon-based approach is much simpler, however, it may lack context understanding and face challenges in specialized domains [57]. The choice between the two approaches depends on the specific requirements and constraints of the SA task at hand.

3. Materials and Methods

To develop code for SA, we recommend installing Python and using a specific extension. To do so, it is necessary to install a Python source code from the official website [58], along with a separate extension, such as Microsoft Visual Studio Code (VS Code). VS Code was chosen due to its widespread popularity and extensive usage as a source code editor. Its comprehensive support for various programming languages offers a wide range of advanced features and extensions. The official website of VS Code [59] provides detailed information on crucial aspects and features, which we have summarized and presented below.

First, VS Code offers cross-platform compatibility, supporting commonly used operating systems such as macOS, Windows, and Linux, making it suitable for service companies. Second, the interface of VS Code is user-friendly, intuitive, and customizable. It includes a sidebar for navigation, a central editing area for code editing, and seamless access to the terminal. The integrated terminal enables developers to execute commands directly within the editor, eliminating the need to switch to an external terminal. This feature significantly enhances productivity. Third, VS Code stands out with its IntelliSense functionality, which provides intelligent suggestions based on variable types, function definitions, and imported modules. Furthermore, IntelliSense offers an intelligent code completion feature that suggests context-aware code snippets, function signatures, and relevant documentation. Fourth, the availability of community-provided extensions is another advantage. These extensions offer support for specific programming languages. For instance, the Live Share extension facilitates real-time collaboration among developers, enabling them to share their development environment, code, and even debugging sessions remotely. Fifth, debugging capabilities in VS Code allow developers to set breakpoints, inspect variables, step through code execution, and analyze program behavior. This feature empowers developers to identify and resolve issues effectively. Overall, Visual Studio Code provides an efficient and effective coding environment for various programming languages and platforms. These features and functionalities justify its widespread adoption.

To enhance the development of our code, we employed the Requests library. This library builds upon the foundation of standard Python HTTP libraries such as urllib, offering a more user-friendly and high-level interface for seamless interaction with HTTP. With Requests, we can effortlessly communicate with web services and retrieve data from servers thanks to its simplicity and intuitive design. Moreover, we utilized TextBlob, a crucial Python library that provides a straightforward and intuitive API for performing various NLP tasks. TextBlob builds upon NLTK and offers a simplified interface for common NLP operations like part-of-speech tagging, noun phrase extraction, and SA. By leveraging TextBlob, we streamlined our NLP tasks and improved their convenience and efficiency.

To test the code, we utilized the URL of the Portal da Queixa [60]. This portal functions as a private digital platform owned by Consumers Trust, a technological startup. Its primary purpose is to facilitate communication between consumers and brands, providing a platform for consumers to express their complaints and fostering dialogue between consumers and brands to address and resolve these issues. The reviews available on the portal are predominantly negative, making it an excellent choice for code testing. After analyzing all 43 pages of the Portal da Queixa, we found that every page exhibited a negative sentiment. This observation aligns with the individual comments we analyzed, further confirming the prevalent negativity. Regarding the reliability of this research, we

 doi:10.20944/preprints202408.1131.v1

provide the code snippets along with corresponding explanations in the following section. These code samples are designed to be easily replicated and tested by the readers of this article.

4. Results and Discussion

The Python code that we present below is one of the most basic foundations that can assist junior IE/managers in developing their own custom coding solutions tailored to their specific organizational needs. The Python code explanation is presented in a step-by-step manner, ensuring thoroughness and clarity for the reader's comprehension. To install Requests and TextBlob using pip, open the terminal or command prompt and run the following command:

```
pip install requests textblob
```

The command will download and install both the Requests and TextBlob packages from the Python Package Index (PyPI) using pip, the Python package manager. Import the necessary libraries by opening the Python script:

```
import requests
from textblob import TextBlob
```

It should be noted that the code utilizes the TextBlob library to perform a straightforward SA task. TextBlob leverages a readily accessible pre-trained model for this purpose. It is important to consider that the accuracy of SA may vary due to two key factors: the quality of the evolving model over time and the characteristics of the analyzed text itself. The code snippet includes two important statements. The provided code snippet includes two import statements that enable the usage of external libraries. The first statement (import Requests) imports the Requests library, which is a commonly used Python library for making HTTP requests to web servers. It offers a straightforward and convenient way to interact with web resources, allowing tasks such as fetching data from URLs and interacting with web APIs. The Requests library simplifies the process of sending HTTP requests and handling responses, making it an essential tool for web-related operations. The second statement (from textblob import TextBlob) imports the textblob class from the TextBlob library. It offers features such as tokenization, which involves splitting text into smaller units (tokens), and SA, which enables the evaluation of the sentiment expressed in a piece of text. The TextBlob library simplifies the implementation of these operations by providing intuitive and easy-to-use interfaces, making it a valuable resource for text analysis tasks. Function to perform SA:

```
def analyze_sentiment (text):
                    blob = TextBlob (text)
                    sentiment = blob.sentiment.polarity
                    if sentiment > 0:
                        return `Positive`
                    elif sentiment < 0:
                        return `Negative`
                    else:
                        return `Neutral`
```

The provided code snippet demonstrates the analyze_sentiment function, which performs SA on a given text. The analyze_sentiment function utilizes the TextBlob library to analyze the sentiment of text input. When invoked, the process accepts a text parameter and creates a blob object using the TextBlob library. This blob object represents the text and enables various natural language processing operations. Furthermore, the function calculates the text's sentiment polarity by accessing the blob object's sentiment—polarity attribute. The sentiment polarity is a numerical value that captures the sentiment of the text, ranging from -1 (indicating a negative sentiment) to 1 (indicating a positive sentiment). Based on the calculated sentiment polarity, the function determines the sentiment label

 doi:10.20944/preprints202408.1131.v1

as follows: If the sentiment polarity is greater than 0, it indicates a positive sentiment, and the function returns the string 'Positive'. If the sentiment polarity is less than 0, it suggests a negative sentiment, and the function returns the string 'Negative'. If the sentiment polarity is exactly 0, it implies a neutral sentiment, and the function returns the string 'Neutral'. In summary, the analyze_sentiment function utilizes the TextBlob library to evaluate the sentiment polarity of a given text and provides an appropriate sentiment label ('Positive', 'Negative', or 'Neutral') based on the calculated polarity. URL of the webpage that is needed to analyze:

```
url = https://portaldaqueixa.com
# Add as many URLs as needed
```

The given code snippet includes a variable named URL, which holds the URL of a selected web page. In this case, the URL is from Portal da Queixa, a website specifically focused on collecting negative complaints from customers who have utilized a particular service. It is important to note that the provided code snippet solely consists of the URL variable, and any additional parsing or operations on the webpage would necessitate further code that is not displayed in the given snippet. GET request to the predefined webpage:

```
response = requests.get(url)
```

The line of code get() function, is part of the Requests library that sends an HTTP GET request to the URL we previously defined, and returned a response. The URL is a variable or string that represents the Uniform Resource Locator (URL) of the resource the IE/manager intends to retrieve or interact with using the HTTP GET method. In the code response = requests.get(url), this line sends an HTTP GET request to the specified URL and assigns the resulting response object to the variable response. This response object contained various details about the server's response, including the status code, headers, and content. After executing this line of code, the IE/manager can access several properties of the response object. For instance, response.status_code retrieves the HTTP status code, response.headers allow access to the response headers and response.text retrieves the response content as a string. These properties and methods provide valuable information for processing the received response from the server. Check if the request was successful by extracting the text content from the response and SA performance:

```
if response.status_code == 200
webpage_content = response.text
```

The line of code if response.status_code == 200 refers to the response object obtained from the earlier HTTP request made using the requests.get() method. The status_code is a property of the response object that holds the HTTP status code of the response. This status code is a three-digit number that provides information about the outcome of the request. While the == operator is an equality comparison operator that checks whether the value on the left is equal to the value on the right. The HTTP status code 200 is a widely used code that signifies a successful response. In the HTTP protocol, a status code of 200 indicates that the request was successful, and the server has responded with the expected data. It represents the "OK" status. Therefore, the line of code if response.status_code == 200 serves as a conditional statement that checks if the status code of the response is equal to 200, indicating a successful response. If the condition evaluates to true, the block of code following the if statement will be executed. This allows to handle the response accordingly or perform specific actions when the request is successful. The second line of code webpage_content is a variable to which we assigned the value of response.text. This variable will hold the content retrieved from the webpage as a string. That is, the line of code webpage_content = response.text retrieves the content of the response and assigns it to the webpage_content variable. The content is obtained as a string, representing the HTML or textual content of the requested web page. After executing this line of code, it is possible to work with the webpage_content variable to process or

 doi:10.20944/preprints202408.1131.v1

parse the retrieved content, such as parsing HTML, which allows SA. Performing sentiment analysis on the selected webpage:

```
sentiment = analyze_sentiment(webpage_content)
```

The line of code sentiment = analyze_sentiment(webpage_content) involves the analyze_sentiment function, which performs SA on the provided input. In this case, the webpage_content variable holds the content of the chosen webpage, obtained through the HTTP request, and stored as a string. The sentiment variable serves as a container for the outcome or output of the analyze_sentiment function. It represents the sentiment analysis performed on the webpage_content. Thus, when executing sentiment = analyze_sentiment(webpage_content), the webpage_content is passed as an argument to the analyze_sentiment function, which evaluates the sentiment expressed within that content. The resulting sentiment is then assigned to the sentiment variable for further utilization. After this line of code, the IE/manager works with the sentiment variable to interpret and utilize the sentiment analysis results.

```
                print (`Sentiment`, sentiment)
else:
                print (`Error:`, response.status_code)
```

In this final section, the code presents the results of the SA. If the analysis is successful, it will display the sentiment (positive, negative, neutral) in the terminal. However, if there is a failure during the execution, it will display an error message. The provided code conducts SA on all the content present on the chosen webpage. If the IE/manager intends to analyze specific sections or elements of the webpage, it will be necessary to modify the code accordingly to accommodate this requirement.

Figure 2 illustrates a general overview of SA. The process begins with the selection of the programming language, source-code editor, and operating system. Following, the required libraries supporting SA need to be installed. The next step involves specifying the function responsible for conducting SA and choosing the target webpage. After confirming the status and extracting the content from the provided URL, SA is performed. Finally, the result is displayed, resulting from the pre-established sentiment function. This flowchart depicts the essential components of sentiment analysis, which can be further enhanced with additional features that are deemed valuable by EI/managers.

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 15 August 2024 doi:10.20944/preprints202408.1131.v1

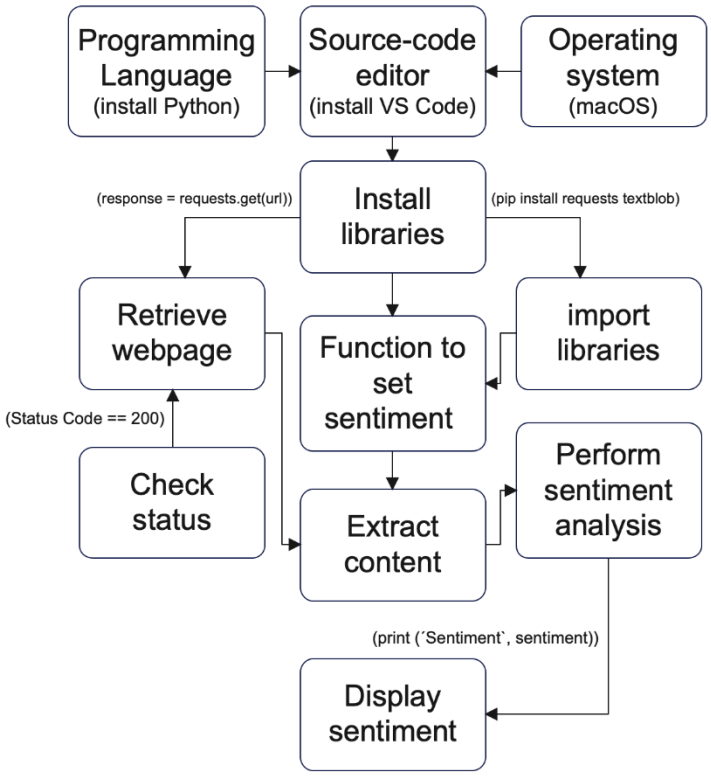


Figure 2. Sentiment Analysis Flowchart.

5. Conclusion

This section focuses on four relevant issues: contributions to theory and management, limitations, and suggestions for future research.

5.1. Theoretical Contributions

This article introduces original theoretical contributions and new resources. Our argument is supported by a preliminary search conducted in Scopus, specifically targeting keywords "python", "sentiment analysis," and "education" in the title, abstract, and keyword fields. As a result, this search yielded 9 journal articles focused on the domains of health [61–63] and social networks [62,64–66]. Hence, this article presents new contributions by introducing Python code for SA, catering specifically to engineers and industrial managers early in their careers or final years of university. These novel contributions have not been published previously, making this article a valuable resource. However, it is important to note that papers such as the one published by Saura et al. [66] explores topic modeling using LDA (latent Dirichlet allocation) to identify topics, which are subsequently subjected to SA using machine learning techniques implemented in Python. The findings of that article offer valuable insights into innovative educational trends that professionals can utilize to enhance strategies and interventions in the education sector. In contrast, our article adopts a lexicon-based approach, providing an alternative methodology for SA to be used for educational or practical purposes.

Potential criticisms of this article center both on the argument that there is already a vast set of tutorials that explain how to perform SA on online platforms such as YouTube, and that the lines of code presented in this article are straightforward. However, this formulation has not yet been described academically and has been increasingly requested by companies and junior IE/managers who want to learn more about the sentiment in service delivery. For this reason, both professionals and students have been forced to consult a wide range of tutorials and too much information on the

 doi:10.20944/preprints202408.1131.v1

subject. Thus, this article intends to fill the void that exists in the literature as it develops the basis for a work aimed at the needs of an audience of young professionals and senior university students.

5.2. Managerial Contributions

This article's primary objective of developing Python code for SA is to empower industrial engineers and managers in service companies to perform SA tasks effectively. By providing a simple but robust codebase, due to popular libraries that build upon NLTK, this article aims to facilitate the implementation of SA techniques in their day-to-day operations. Moreover, a significant contribution to management lies in inspiring junior IE/managers to explore and create their own customized coding solutions, aligned with their organization's unique requirements. Encouraging the development of bespoke solutions empowers managers to address specific challenges and leverage the potential of SA in their respective domains. This approach fosters innovation and ensures that SA is seamlessly integrated into the organizational workflow, maximizing its impact on decision-making processes. Finally, the findings of this article are currently being effectively implemented in classes on Engineering and Process Management in Services, specifically in the master's program for Engineering and Industrial Management at Lusófona University in Portugal. The student's response has been overwhelmingly positive, with excellent feedback received. The success of this implementation can be attributed to the application of Python programming, which facilitates the integration of theoretical knowledge in service science with practical management skills required in real-world business scenarios.

5.3. Limitations and Suggestions for Future Research

Conducting research in SA, like any other field, comes with its own set of limitations. Acknowledging that SA can exhibit variations across different domains and sectors is essential. Models trained in one domain may not possess the capability to generalize effectively to diverse domains. This limitation emphasizes the need for domain-specific adaptation [67] and fine-tuning [68] to achieve optimal performance in sentiment analysis. Another significant challenge in sentiment analysis research is determining the ground truth sentiment labels for text data [69]. This task often involves subjective judgment since human annotators may disagree on the sentiment polarity of certain texts. The absence of a definitive and universally agreed-upon standard for sentiment annotation further complicates the evaluation and comparison of different sentiment analysis approaches. Additionally, language itself undergoes constant evolution due to cultural changes, societal shifts, and the introduction of new vocabularies. This evolving nature necessitates continuous adaptation and updates in sentiment analysis models to accurately capture and interpret the sentiment of current language usage. Models trained on older data may struggle to keep pace with the dynamic linguistic landscape, potentially impacting their performance and applicability. To overcome these limitations, researchers in sentiment analysis need to emphasize domain-specific adaptation, establish standardized annotation guidelines and protocols, and regularly update their models to reflect the changing nature of language. By addressing these challenges, researchers can enhance the reliability and effectiveness of sentiment analysis techniques across various domains and ensure their applicability in real-world scenarios.

Moving forward, our research will concentrate on advancing the development of web applications, SA APIs (Application Programming Interfaces), and AI applications. These advancements aim to enable the creation of tailored code solutions for companies, fostering seamless collaborations between humans and machines. By focusing on web applications, we seek to create user-friendly interfaces that allow businesses to harness the power of SA efficiently. These applications will provide intuitive features and functionalities, enabling users to leverage SA techniques without extensive coding knowledge. Additionally, the development of SA APIs will offer a standardized and accessible way for companies to integrate SA capabilities into their existing systems and workflows. These APIs will provide well-defined interfaces and methods, simplifying the process of incorporating SA functionalities into various applications. Furthermore, our research will explore the potential of AI applications in the realm of SA. By leveraging artificial intelligence

and machine learning techniques, we aim to develop intelligent systems that can automatically analyze sentiment, interpret data, and provide valuable insights to businesses. Overall, we believe that these future developments will play a pivotal role in promoting the creation of tailored code solutions for companies. By facilitating human-machine collaborations, we aim to empower organizations to make informed decisions based on accurate sentiment analysis, driving innovation, and improving overall performance.

Author Contributions: Conceptualization, J.R.; methodology, J.R.; software, J.R.; validation, J.R.; formal analysis, J.R.; investigation, J.R.; resources, J.R.; data curation, J.R.; writing—original draft preparation, J.R.; writing—review and editing, J.R.; visualization, J.R.; supervision, J.R.; project administration, J.R.; funding acquisition, J.R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. L. Collins, P. D. Hannon, e A. Smith, «Enacting entrepreneurial intent: the gaps between student needs and higher education capability», *Educ. Train.*, vol. 46, n.º 8/9, pp. 454–463, out. 2004, DOI: 10.1108/00400910410569579.
2. M. Barrett e E. Davidson, «Exploring the Diversity of Service Worlds in the Service Economy», In *Information Technology in the Service Economy: Challenges and Possibilities for the 21st Century*, vol. 267, M. Barrett, E. Davidson, C. Middleton, e J. I. DeGross, Eds., In IFIP — The International Federation for Information Processing, vol. 267, Boston, MA: Springer US, 2008, pp. 1–10. DOI: 10.1007/978-0-387-09768-8_1.
3. D. Soto Setzke, T. Riasanow, M. Böhm, e H. Krcmar, «Pathways to Digital Service Innovation: The Role of Digital Transformation Strategies in Established Organizations», *Inf. Syst. Front.*, vol. 25, n.º 3, pp. 1017–1037, jun. 2023, DOI: 10.1007/s10796-021-10112-0.
4. J. Reis e N. Melão, «Digital transformation: A meta-review and guidelines for future research», *Heliyon*, vol. 9, n.º 1, p. e12834, jan. 2023, DOI: 10.1016/j.heliyon.2023.e12834.
5. S. Gupta, W. Ghardallou, D. K. Pandey, e G. P. Sahu, «Artificial intelligence adoption in the insurance industry: Evidence using the technology–organization–environment framework», *Res. Int. Bus. Finance*, vol. 63, p. 101757, dez. 2022, DOI: 10.1016/j.ribaf.2022.101757.
6. P. C. Marques, J. Reis, e R. Santos, «Artificial Intelligence and Disruptive Technologies in Service Systems: A Bibliometric Analysis», *Int. J. Innov. Technol. Manag.*, p. 2330003, jun. 2023, DOI: 10.1142/S0219877023300033.
7. R. Maruta, «The creation and management of organizational knowledge», *Knowl.-Based Syst.*, vol. 67, pp. 26–34, set. 2014, DOI: 10.1016/j.knosys.2014.06.012.
8. A. Kolmos, R. G. Hadgraft, e J. E. Holgaard, «Response strategies for curriculum change in engineering», *Int. J. Technol. Des. Educ.*, vol. 26, n.º 3, pp. 391–411, ago. 2016, DOI: 10.1007/s10798-015-9319-y.
9. A. Benis, S. Amador Nelke, e M. Winokur, «Training the Next Industrial Engineers and Managers about Industry 4.0: A Case Study about Challenges and Opportunities in the COVID-19 Era», *Sensors*, vol. 21, n.º 9, p. 2905, abr. 2021, DOI: 10.3390/s21092905.
10. J. Reis, P. E. Santo, e N. Melão, «Artificial Intelligence in Government Services: A Systematic Literature Review», In *New Knowledge in Information Systems and Technologies*, vol. 930, Á. Rocha, H. Adeli, L. P. Reis, e S. Costanzo, Eds., In Advances in Intelligent SystIns and Computing, vol. 930. Cham: Springer International Publishing, 2019, pp. 241–252. DOI: 10.1007/978-3-030-16181-1_23.
11. Sambamurthy, Bharadwaj, e Grover, «Shaping Agility through Digital Options: Reconceptualizing the Role of Information Technology in Contemporary Firms», *MIS Q.*, vol. 27, n.º 2, p. 237, 2003, DOI: 10.2307/30036530.
12. W. He, H. Wu, G. Yan, V. Akula, e J. Shen, «A novel social media competitive analytics framework with sentiment benchmarks», *Inf. Manage.*, vol. 52, n.º 7, pp. 801–812, nov. 2015, DOI: 10.1016/j.im.2015.04.006.
13. J. Serrano-Guerrero, J. A. Olivas, F. P. Romero, e E. Herrera-Viedma, «Sentiment analysis: A review and comparative analysis of web services», *Inf. Sci.*, vol. 311, pp. 18–38, ago. 2015, DOI: 10.1016/j.ins.2015.03.040.
14. S. Schmunk, W. Höpken, M. Fuchs, e M. Lexhagen, «Sentiment Analysis: Extracting Decision-Relevant Knowledge from UGC», In *Information and Communication Technologies in Tourism 2014*, Z. Xiang e I. Tussyadiah, Eds., Cham: Springer International Publishing, 2013, pp. 253–265. DOI: 10.1007/978-3-319-03973-2_19.
15. Z. Wang, L. Wang, Y. Ji, L. Zuo, e S. Qu, «A novel data-driven weighted sentiment analysis based on information entropy for perceived satisfaction», *J. Retail. Consum. Serv.*, vol. 68, p. 103038, set. 2022, DOI: 10.1016/j.jretconser.2022.103038.

16. D. N. Mishra e R. K. Panda, «Decoding customer experiences in rail transport service: application of hybrid sentiment analysis», *Public Transp.*, vol. 15, n.º 1, pp. 31–60, mar. 2023, DOI: 10.1007/s12469-021-00289-7.
17. S. Farzadnia e I. Raeesi Vanani, «Identification of opinion trends using sentiment analysis of airlines passengers' reviews», *J. Air Transp. Manag.*, vol. 103, p. 102232, ago. 2022, DOI: 10.1016/j.jairtraman.2022.102232.
18. S.-W. Lee, G. Jiang, H.-Y. Kong, e C. Liu, «A difference of multimedia consumer's rating and review through sentiment analysis», *Multimed. Tools Appl.*, vol. 80, n.º 26–27, pp. 34625–34642, nov. 2021, DOI: 10.1007/s11042-020-08820-x.
19. J. Jussila, V. Vuori, J. Okkonen, e N. Helander, «Reliability and Perceived Value of Sentiment Analysis for Twitter Data», In *Strategic Innovative Marketing*, A. Kavoura, D. P. Sakas, e P. Tomaras, Eds., In Springer Proceedings in Business and Economics. , Cham: Springer International Publishing, 2017, pp. 43–48. DOI: 10.1007/978-3-319-56288-9_7.
20. M. McGuire e C. Kampf, «Using social media sentiment analysis to understand audiences: A new skill for technical communicators? In *2015 IEEE International Professional Communication Conference (IPCC)*, Limerick, Ireland: IEEE, jul. 2015, pp. 1–7. DOI: 10.1109/IPCC.2015.7235801.
21. S. Bird, E. Klein, e E. Loper, *Natural language processing with Python*, 1st ed. Beijing; Cambridge [Mass.]: O'Reilly, 2009.
22. E. Loper e S. Bird, «NLTK: The Natural Language Toolkit», 2002, DOI: 10.48550/ARXIV.CS/0205028.
23. J. Wang, B. Xu, e Y. Zu, «Deep learning for Aspect-based Sentiment Analysis», In *2021 International Conference on Machine Learning and Intelligent Systems Engineering (MLISE)*, Chongqing, China: IEEE, jul. 2021, pp. 267–271. DOI: 10.1109/MLISE54096.2021.00056.
24. S. Behdenna, F. Barigou, e G. Belalem, «Sentiment Analysis at Document Level», In *Smart Trends in Information Technology and Computer Communications*, vol. 628, A. Unal, M. Nayak, D. K. Mishra, D. Singh, e A. Joshi, Eds., In Communications in Computer and Information Science, vol. 628. , Singapore: Springer Singapore, 2016, pp. 159–168. DOI: 10.1007/978-981-10-3433-6_20.
25. W. Medhat, A. Hassan, e H. Korashy, «Sentiment analysis algorithms and applications: A survey», *Ain Shams Eng. J.*, vol. 5, n.º 4, pp. 1093–1113, dez. 2014, DOI: 10.1016/j.asej.2014.04.011.
26. S. Park, S. Strover, J. Choi, e M. Schnell, «Mind games: A temporal sentiment analysis of the political messages of the Internet Research Agency on Facebook and Twitter», *New Media Soc.*, vol. 25, n.º 3, pp. 463–484, mar. 2023, DOI: 10.1177/14614448211014355.
27. S. E. Bestvater e B. L. Monroe, «Sentiment is Not Stance: Target-Aware Opinion Classification for Political Text Analysis», *Polit. Anal.*, vol. 31, n.º 2, pp. 235–256, abr. 2023, DOI: 10.1017/pan.2022.10.
28. Y. Xu e V. Keselj, «Stock Prediction using Deep Learning and Sentiment Analysis», In *2019 IEEE International Conference on Big Data (Big Data)*, Los Angeles, CA, USA: IEEE, dez. 2019, pp. 5573–5580. DOI: 10.1109/BigData47090.2019.9006342.
29. F. J. Ramírez-Tinoco, G. Alor-Hernández, J. L. Sánchez-Cervantes, M. D. P. Salas-Zárate, e R. Valencia-García, «Use of Sentiment Analysis Techniques in Healthcare Domain», In *Current Trends in Semantic Web Technologies: Theory and Practice*, vol. 815, G. Alor-Hernández, J. L. Sánchez-Cervantes, A. Rodríguez-González, e R. Valencia-García, Eds., In Studies in Computational Intelligence, vol. 815. , Cham: Springer International Publishing, 2019, pp. 189–212. DOI: 10.1007/978-3-030-06149-4_8.
30. K. Denecke e Y. Deng, «Sentiment analysis in medical settings: New opportunities and challenges», *Artif. Intell. Med.*, vol. 64, n.º 1, pp. 17–27, mai. 2015, DOI: 10.1016/j.artmed.2015.03.006.
31. E. Haddi, X. Liu, e Y. Shi, «The Role of Text Pre-processing in Sentiment Analysis», *Procedia Comput. Sci.*, vol. 17, pp. 26–32, 2013, DOI: 10.1016/j.procs.2013.05.005.
32. M. Taboada, J. Brooke, M. Tofiloski, K. Voll, e M. Stede, «Lexicon-Based Methods for Sentiment Analysis», *Comput. Linguist.*, vol. 37, n.º 2, pp. 267–307, jun. 2011, DOI: 10.1162/COLI_a_00049.
33. M. S. Neethu e R. Rajasree, «Sentiment analysis in twitter using machine learning techniques», In *2013 Fourth International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode: IEEE, jul. 2013, pp. 1–5. DOI: 10.1109/ICCCNT.2013.6726818.
34. A. Das, S. Bandyopadhyay, e B. Gambäck, «Sentiment analysis: what is the end user's requirement? In *Proceedings of the 2nd International Conference on Web Intelligence, Mining and Semantics*, Craiova Romania: ACM, jun. 2012, pp. 1–10. DOI: 10.1145/2254129.2254173.
35. M. Javed e S. Kamal, «Normalization of Unstructured and Informal Text in Sentiment Analysis», *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, n.º 10, 2018, DOI: 10.14569/IJACSA.2018.091011.
36. T. Nasukawa e J. Yi, «Sentiment analysis: capturing favorability using natural language processing», In *Proceedings of the 2nd international conference on Knowledge capture*, Sanibel Island FL USA: ACM, out. 2003, pp. 70–77. DOI: 10.1145/945645.945658.
37. A. Kamath, R. Guhekar, M. Makwana, e S. N. Dhage, «Sarcasm Detection Approaches Survey», In *Advances in Computer, Communication and Computational Sciences*, vol. 1158, S. K. Bhatia, S. Tiwari, S. Ruidan, M. C. Trivedi, e K. K. Mishra, Eds., In Advances in Intelligent Systems and Computing, vol. 1158. , Singapore: Springer Singapore, 2021, pp. 593–609. DOI: 10.1007/978-981-15-4409-5_54.

38. A. Balahur, R. Mihalcea, e A. Montoyo, «Computational approaches to subjectivity and sentiment analysis: Present and envisaged methods and applications», *Comput. Speech Lang.*, vol. 28, n.º 1, pp. 1–6, jan. 2014, DOI: 10.1016/j.csl.2013.09.003.
39. K. Ravi e V. Ravi, «A survey on opinion mining and sentiment analysis: Tasks, approaches and applications», *Knowl.-Based Syst.*, vol. 89, pp. 14–46, nov. 2015, DOI: 10.1016/j.knosys.2015.06.015.
40. Gang Li e Fei Liu, «A clustering-based approach on sentiment analysis», In *2010 IEEE International Conference on Intelligent Systems and Knowledge Engineering*, Hangzhou, China: IEEE, nov. 2010, pp. 331–337. DOI: 10.1109/ISKE.2010.5680859.
41. Y. He e D. Zhou, «Self-training from labeled features for sentiment analysis», *Inf. Process. Manag.*, vol. 47, n.º 4, pp. 606–616, jul. 2011, DOI: 10.1016/j.ipm.2010.11.003.
42. S. Pradha, M. N. Halgamuge, e N. Tran Quoc Vinh, «Effective Text Data Preprocessing Technique for Sentiment Analysis in Social Media Data», In *2019 11th International Conference on Knowledge and Systems Engineering (KSE)*, Da Nang, Vietnam: IEEE, out. 2019, pp. 1–8. DOI: 10.1109/KSE.2019.8919368.
43. R. Ahuja, A. Chug, S. Kohli, S. Gupta, e P. Ahuja, «The Impact of Features Extraction on the Sentiment Analysis», *Procedia Comput. Sci.*, vol. 152, pp. 341–348, 2019, DOI: 10.1016/j.procs.2019.05.008.
44. M. Avinash e E. Sivasankar, «A Study of Feature Extraction Techniques for Sentiment Analysis», In *Emerging Technologies in Data Mining and Information Security*, vol. 814, A. Abraham, P. Dutta, J. K. Mandal, A. Bhattacharya, e S. Dutta, Eds., In Advances in Intelligent Systems and Computing, vol. 814. , Singapore: Springer Singapore, 2019, pp. 475–486. DOI: 10.1007/978-981-13-1501-5_41.
45. K. Baktha e B. K. Tripathy, «Investigation of recurrent neural networks in the field of sentiment analysis», In *2017 International Conference on Communication and Signal Processing (ICCSP)*, Chennai: IEEE, abr. 2017, pp. 2047–2050. DOI: 10.1109/ICCSP.2017.8286763.
46. S. Sachin, A. Tripathi, N. Mahajan, S. Aggarwal, e P. Nagrath, «Sentiment Analysis Using Gated Recurrent Neural Networks», *SN Comput. Sci.*, vol. 1, n.º 2, p. 74, mar. 2020, DOI: 10.1007/s42979-020-0076-y.
47. H. Fei, T.-S. Chua, C. Li, D. Ji, M. Zhang, e Y. Ren, «On the Robustness of Aspect-based Sentiment Analysis: Rethinking Model, Data, and Training», *ACM Trans. Inf. Syst.*, vol. 41, n.º 2, pp. 1–32, abr. 2023, DOI: 10.1145/3564281.
48. M. Wankhade, A. C. S. Rao, e C. Kulkarni, «A survey on sentiment analysis methods, applications, and challenges», *Artif. Intell. Rev.*, vol. 55, n.º 7, pp. 5731–5780, out. 2022, DOI: 10.1007/s10462-022-10144-1.
49. S. Jindal e S. Singh, «Image sentiment analysis using deep convolutional neural networks with domain specific fine tuning», In *2015 International Conference on Information Processing (ICIP)*, Pune, India: IEEE, dez. 2015, pp. 447–451. DOI: 10.1109/INFOP.2015.7489424.
50. N. Mukhtar, M. A. Khan, e N. Chiragh, «Lexicon-based approach outperforms Supervised Machine Learning approach for Urdu Sentiment Analysis in multiple domains», *Telemat. Inform.*, vol. 35, n.º 8, pp. 2173–2183, dez. 2018, DOI: 10.1016/j.tele.2018.08.003.
51. F. Zarisfi Kermani, F. Sadeghi, e E. Eslami, «Solving the twitter sentiment analysis problem based on a machine learning-based approach», *Evol. Intell.*, vol. 13, n.º 3, pp. 381–398, set. 2020, DOI: 10.1007/s12065-019-00301-x.
52. H.-T. Duong e T.-A. Nguyen-Thi, «A review: preprocessing techniques and data augmentation for sentiment analysis», *Comput. Soc. Netw.*, vol. 8, n.º 1, p. 1, dez. 2021, DOI: 10.1186/s40649-020-00080-x.
53. C. Dhaoui, C. M. Webster, e L. P. Tan, «Social media sentiment analysis: lexicon versus machine learning», *J. Consum. Mark.*, vol. 34, n.º 6, pp. 480–488, set. 2017, DOI: 10.1108/JCM-03-2017-2141.
54. J. Yi, T. Nasukawa, R. Bunescu, e W. Niblack, «Sentiment analyzer: extracting sentiments about a given topic using natural language processing techniques», In *Third IEEE International Conference on Data Mining*, Melbourne, FL, USA: IEEE Comput. Soc, 2003, pp. 427–434. DOI: 10.1109/ICDM.2003.1250949.
55. A. Moreo, M. Romero, J. L. Castro, e J. M. Zurita, «Lexicon-based Comments-oriented News Sentiment Analyzer system», *Expert Syst. Appl.*, vol. 39, n.º 10, pp. 9166–9180, ago. 2012, DOI: 10.1016/j.eswa.2012.02.057.
56. A. Hogenboom, B. Heerschop, F. Frasincar, U. Kaymak, e F. De Jong, «Multi-lingual support for lexicon-based sentiment analysis guided by semantics», *Decis. Support Syst.*, vol. 62, pp. 43–53, jun. 2014, DOI: 10.1016/j.dss.2014.03.004.
57. F. Chiarello, A. Bonaccorsi, e G. Fantoni, «Technical Sentiment Analysis. Measuring Advantages and Drawbacks of New Products Using Social Media», *Comput. Ind.*, vol. 123, p. 103299, dez. 2020, DOI: 10.1016/j.compind.2020.103299.
58. Python. Available online: https://www.python.org/downloads/ (accessed on 11, June 2023).
59. Visual Studio Code. Available online: https://code.visualstudio.com (accessed on 11, June 2023).
60. Portal da Queixa. Available online: hhttps://portaldaqueixa.com (accessed on 17, June 2023).
61. B. W. Pang *et al.*, «Innovative growth and development of a neurological surgery residency cadaveric skull base simulation training program: A single institution experience», *Clin. Neurol. Neurosurg.*, vol. 225, p. 107585, fev. 2023, DOI: 10.1016/j.clineuro.2023.107585.

 doi:10.20944/preprints202408.1131.v1

62. S. Park e Y.-K. Suh, «A Comprehensive Analysis of COVID-19 Vaccine Discourse by Vaccine Brand on Twitter in Korea: Topic and Sentiment Analysis», *J. Med. Internet Res.*, vol. 25, p. e42623, jan. 2023, DOI: 10.2196/42623.
63. Q. Liu *et al.*, «Health Communication Through News Media During the Early Stage of the COVID-19 Outbreak in China: Digital Topic Modeling Approach», *J. Med. Internet Res.*, vol. 22, n.º 4, p. e19118, abr. 2020, DOI: 10.2196/19118.
64. N. Kewsuwun e S. Kajornkasirat, «A sentiment analysis model of Agritech startup on Facebook comments using naive Bayes classifier», *Int. J. Electr. Comput. Eng. IJECE*, vol. 12, n.º 3, p. 2829, jun. 2022, DOI: 10.11591/ijece.v12i3.pp2829-2838.
65. D. Leitch e M. Sherif, «Twitter mood, CEO succession announcements and stock returns», *J. Comput. Sci.*, vol. 21, pp. 1–10, jul. 2017, DOI: 10.1016/j.jocs.2017.04.002.
66. J. R. Saura, A. Reyes-Menendez, e D. R. Bennett, «How to Extract Meaningful Insights from UGC: A Knowledge-Based Method Applied to Education», *Appl. Sci.*, vol. 9, n.º 21, p. 4603, out. 2019, DOI: 10.3390/app9214603.
67. Y. Choi, Y. Kim, e S.-H. Myaeng, «Domain-specific sentiment analysis using contextual feature generation», In *Proceedings of the 1st international CIKM workshop on Topic-sentiment analysis for mass opinion*, Hong Kong China: ACM, nov. 2009, pp. 37–44. DOI: 10.1145/1651461.1651469.
68. N. J. Prottasha *et al.*, «Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning», *Sensors*, vol. 22, n.º 11, p. 4157, mai. 2022, DOI: 10.3390/s22114157.
69. P. Gonçalves, M. Araújo, F. Benevenuto, e M. Cha, «Comparing and combining sentiment analysis methods», In *Proceedings of the first ACM conference on Online social networks*, Boston Massachusetts USA: ACM, out. 2013, pp. 27–38. DOI: 10.1145/2512938.2512951.