2023 International Conference on Artificial Intelligence and its Applications
https://doi.org/10.59200/ICARTI.2023.028

Exploring Real-Time Sentiment Analysis Prototype for Retail Industry

William Chong Wo Yuen
Dodo Technologies Ltd,
William-chong@outlook.com

Suraj Juddoo
School of Science and Technology,
Middlesex University Mauritius,
Unici,Flic en Flac, Mauritius.
s.juddoo@mdx.ac.mu

***Abstract*—The rise of social media platforms has revolutionized the way consumers interact with retailers and express their opinions on products and services. Online retailers particularly need to keep a close eye on customer sentiment in real-time to make informed decisions about their offerings and improve customer satisfaction. However, efficiently analysing large volumes of unstructured text data from social media in real-time poses a significant challenge. This research aimed to develop a scalable, real-time sentiment analysis system tailored for online retailers using Reddit as the data source. The system comprises three main components: a data extraction and streaming pipeline, a sentiment analysis model, and a web application with real-time analytics. To address the data extraction challenge, a job queue-based system was implemented using Node.js, 'BullMQ', and Redis to create and manage campaigns for data streaming from Reddit. The data was streamed using Kafka, a distributed streaming platform, to enable efficient real-time processing. The sentiment analysis model was developed using a Naive Bayes classifier after experimenting with other machine learning and deep learning techniques. In the conducted study, the sentiment analysis model's performance was evaluated using standard metrics tailored to the context of online retail sentiment analysis. An accuracy of 0.6737 was achieved, reflecting the model's ability to correctly classify approximately 67.37 per cent of the sentiments in the test data. Concurrently, an F1 score of 0.7894 was recorded and the Area Under the Curve (AUC) value on the test data was measured at 0.5468, a metric that, while acceptable, suggests room for further refinement in the model's discriminatory ability between classes. The integration of the Data Version Control (DVC) system provided a mechanism for fine-tuning the model according to specific data requirements of various tenants. These results, taken together, not only validate the feasibility of employing a Naive Bayes classifier for real-time sentiment analysis in the retail context, but also provide a baseline for future research aimed at enhancing both the accuracy and efficiency of sentiment classification. The project's evaluation focused on the performance of the sentiment analysis model, the efficiency of the Kafka streaming and real-time Spark pre-processing pipeline, and the backend infrastructure, including the job queuing system and WebSocket implementation. Various evaluation techniques, such as graphs and literature comparisons, were used to assess the system's performance. In conclusion, this project successfully demonstrated the feasibility of a scalable, real-time sentiment analysis system for online retailers using Reddit data. The system has the potential to help retailers better understand customer opinions and make data-driven decisions for their businesses. Future work could include exploring alternative data sources, experimenting with more advanced sentiment analysis techniques, and enhancing the web application's user interface and analytics capabilities.**

I. INTRODUCTION

In the retail industry, where online conversations and social media platforms play an increasingly critical role in shaping consumer opinions and purchasing decisions, sentiment analysis has become a pivotal tool for businesses. The traditional methods of capturing customer sentiment, such as surveys and manual analysis of customer reviews, have proven inadequate to meet the real-time demands of today's retail landscape [14].

With the proliferation of platforms such as Twitter, Facebook, and Reddit, customers are freely expressing their thoughts and feelings about products and services [24]. These platforms generate an enormous volume of data, rich in authentic sentiment, yet challenging to harness effectively. Existing sentiment analysis systems have found it increasingly difficult to keep pace with the speed, volume, and complexity of this data [28]. They often struggle with accuracy, real-time processing, and adaptability, especially when considering the multifaceted nature of a multi-tenant environment.

The ability to analyse sentiment in real time provides retailers with immediate insights into customer preferences, trends, and potential areas for improvement. It not only allows them to respond quickly to market dynamics, but also to tailor their offerings more precisely to consumer needs. However, the sheer diversity and scale of online conversations necessitate more advanced, scalable, and real-time sentiment analysis systems specifically tailored for the retail industry [1].

The problem is further compounded in a multi-tenant context. In sentiment analysis, "tenants" refer to distinct entities within a system, such as individual retailers or larger organizations, each operating within isolated environments, with unique data requirements and permissions. The traditional methods are even more limited in such contexts, failing to capture the ever-changing dynamics and failing to scale effectively.

The urgent need for innovation in this field, coupled with the specific challenges of multi-tenancy, underscores the necessity for a robust, flexible, and real-time sentiment analysis system in modern retail. Such a system should not only adapt to the unique needs of each tenant, but also handle the massive and heterogeneous data generated online, enabling retailers to tap into the authentic sentiments of their customer base, thus driving informed and timely business decisions.

To address these multifaceted challenges, this project introduces an innovative approach that utilizes Data Version Control (DVC) and a robust job queuing system. The integration of DVC ensures that the sentiment analysis models are adaptable to the specific needs of each tenant. By providing a flexible framework for versioning data–and models, DVC allows individual tenants to fine-tune and adapt the system to their unique requirements. This guarantees that sentiment analysis is not only targeted and relevant, but also continually aligned with the evolving data needs and market conditions of each tenant.

Complementing the adaptability provided by DVC, the incorporation of a robust queuing system adds a layer of efficiency and scalability to the sentiment analysis process. By managing the requests and processing from multiple tenants systematically, the queuing system ensures that each tenant's demands are handled promptly and fairly. This creates a seamless experience, even when dealing with high volumes of data from various sources, thus allowing real-time sentiment analysis across different tenants without compromising responsiveness.

In combining DVC with a queuing system, this project presents a novel solution to the multi-tenancy issue in real-time sentiment analysis. It acknowledges the unique characteristics of each tenant, providing a system that is both customizable and scalable. The flexibility to adapt to individual needs, along with the capability to handle diverse data streams in real time, positions this approach at the forefront of modern retail sentiment analysis. It offers a pathway to more genuine and timely insights into customer feelings, trends, and concerns, tailored to the specific context of each tenant. This is a crucial step towards a more connected, agile, and responsive retail industry, leveraging the vast opportunities presented by online platforms and social media channels.

II. LITERATURE REVIEW

In this literature review, the relevant literature in the areas of sentiment analysis techniques, real-time data processing, and the integration of technologies such as Kafka and Spark for building scalable, real-time sentiment analysis systems are explored. The importance of model management and scalability in the context of large-scale sentiment analysis systems are examined. The insights gained from this literature review help guide the design and implementation of our system, as well as inform future research directions in this domain.

A. Rule-Based Approaches

Rule-based sentiment analysis techniques rely on a set of manually crafted rules, lexicons, or patterns to identify sentiments in text data [34]. These approaches typically involve the use of sentiment lexicons, which are dictionaries containing words or phrases along with their associated sentiment scores (positive, negative, or neutral). Some well-known sentiment lexicons include 'SentiWordNet' [11], VADER [17], and LIWC [21].

While rule-based approaches are easy to implement and can produce interpretable results, their performance is often limited by the quality and coverage of the sentiment lexicons used. These approaches also struggle with handling the nuances of natural language, such as sarcasm, idiomatic expressions, and context-dependent sentiment [21].

B. Machine Learning Approaches

Machine learning-based sentiment analysis techniques rely on training models using labelled data to classify text as positive, negative, or neutral [11]. These approaches can be further divided into supervised learning, unsupervised learning, and semi-supervised learning methods [8].

Supervised learning methods, such as Naive Bayes, Support Vector Machines (SVM), and Decision Trees, have been widely used for sentiment analysis [26]. These methods require large amounts of labelled training data, which can be time-consuming and costly to obtain.

Deep learning methods, such as Convolutional Neural Networks (CNN) and Recurrent Neural Networks (RNN), have gained popularity in recent years due to their ability to capture complex relationships and dependencies in text data [19]. These methods have achieved state-of-the-art performance in various sentiment analysis tasks, but they often require large amounts of training data and computational resources [3].

As the size of the training data increases, the accuracy of the model tends to improve. However, there may be diminishing returns in accuracy beyond a certain point, and the cost of obtaining and processing additional data may become prohibitive. These trade-offs should be carefully considered when designing a machine learning-based sentiment analysis system. Unsupervised learning methods, such as clustering and topic modelling, can be used to analyse sentiments in the absence of labelled data [25]. However, these methods often produce less accurate results compared to supervised and deep learning methods, and their applicability to specific sentiment analysis tasks may be limited [8].

C. Apache Kafka

Apache Kafka is a distributed streaming platform that is designed for high-throughput, fault-tolerant, and scalable data streaming [10]. Kafka is widely used for building real-time data pipelines and streaming applications, such as log aggregation, event-driven architectures, and stream processing [22]. Some key features of Kafka include a publish-subscribe model, fault tolerance, and horizontal scalability, which make it suitable for large-scale, real-time data processing tasks [20].

Kafka has been adopted by various companies to address their real-time data processing needs. For instance, Uber uses Kafka to manage the high volume of events generated by its ride-sharing platform, such as driver location updates, ride requests, and fare calculations. Kafka enables Uber to process these events in real-time and maintain a highly responsive system [13].

D. Apache Spark

Apache Spark is an open-source, distributed computing system that provides a unified framework for big data processing, including batch processing, interactive queries, streaming, and machine learning [36]. Spark's in-memory processing capabilities, combined with its support for fault-tolerant data processing, make it well-suited for real-time sentiment analysis tasks [27].

Spark Streaming is a component of Spark that enables the processing of real-time data streams. It divides the input data into small batches and processes them using Spark's core engine, which allows for the same code to be used for both batch processing and real-time streaming applications [36]. Several studies have demonstrated the effectiveness of using Spark Streaming for real-time sentiment analysis tasks, including Twitter data analysis [32] and online review analysis [5].

E. Model Management and Scalability

As the volume of data and the complexity of machine learning models continue to grow, there is an increasing need for systems that can manage and scale machine learning models efficiently. Data Version Control (DVC) is an open-source tool that enables version control and collaboration for machine learning projects [31]. DVC allows users to track changes to their datasets and models, share and collaborate on model development, and reproduce experiments easily [31].

In the context of large-scale sentiment analysis systems, DVC can be used to manage and share models across multiple tenants, allowing for the fine-tuning of models based on individual data requirements. This can help ensure that the system remains scalable and adaptable as new data and requirements emerge.

III. RESEARCH METHODOLOGY

In this section, the research process employed is outlined. To achieve the research goal, an experimental approach was carried out and consisted of the following main steps: (1) the 'BullMQ' job queue was implemented for managing campaign data streaming tasks, providing a robust and efficient solution for handling multiple tenants (2), for data collection, a labelled dataset of comments and reviews related to retailers' brands was obtained (3), and the Naive Bayes classifier was chosen for the sentiment analysis model due to its simplicity and suitability for real-time processing. However, it is worth noting that Support Vector Machines (SVM) was also explored during the project.

After collecting and pre-processing the data, they were processed by machine learning techniques. The accuracy of the sentiment analysis model was evaluated. The choice of steps followed was justified by the need for a real-time, scalable, and accurate sentiment analysis system.

A. High-level Architecture

The high-level architecture proposed for the real-time sentiment analysis system is designed to ensure efficient data processing, seamless component interaction, and scalability. The architecture is composed of several interconnected components, each responsible for a specific aspect of the system's functionality.

1. Data Extraction Component: This component is responsible for extracting relevant data from Reddit threads. The data extraction process uses Reddit's API to stream comments.

2. Data Streaming Component: The data streaming component uses Apache Kafka to manage the flow of data from the extraction component to the processing component. Apache Kafka enables efficient and fault-tolerant data streaming, ensuring that large volumes of data can be handled in real-time.

3. Data Processing Component: The data processing component leverages Apache Spark to perform necessary transformations and prepare the data for sentiment analysis. Spark's in-memory data processing capabilities allow for rapid processing and analysis, facilitating real-time sentiment scoring.

4. Sentiment Analysis Component: This component employs machine learning algorithms and natural language processing techniques to evaluate and quantify the sentiment scores of the extracted data. The sentiment analysis model is trained on a labelled reviews dataset.

5. Data Storage Component: The processed data and sentiment analysis results are stored in Postgres to ensure that data is readily available for retrieval and further analysis.

6. User Interface Component: The user interface component provides a user-friendly dashboard or web application that allows users to view real-time sentiment analysis results and interact with the system. The interface presents data in a visually appealing and comprehensive format, enabling users to easily interpret and analyse sentiment trends.

B. Data Pipeline Design

The focus of the data pipeline design is to create a robust and scalable pipeline capable of handling real-time data from Reddit, processing it efficiently, and delivering the results to the user interface. The primary goal is to ensure the seamless flow of data from the source to the end-user while maintaining system performance and ensuring fault tolerance.

To achieve this, Apache Kafka was chosen as our data streaming platform for its high throughput, fault tolerance, and horizontal scalability, making it an ideal choice for large-scale real-time data processing tasks [4]. By utilizing Kafka's publish-subscribe, the flow of data between different components of our system becomes manageable. This allows for a separate data ingestion, processing, and delivery, making the pipeline more modular and easier to maintain.

The pipeline starts with ingesting comments from a designated Reddit thread using the 'Snoowrap' API. After ingesting the data, it is then sent to Kafka, where it is stored in a topic. Kafka's ability to handle multiple topics simultaneously [13] allows our system to easily scale to accommodate more data sources, or process data from multiple Reddit threads concurrently.

To process the data and perform sentiment analysis, the pipeline was integrated with Apache Spark. Spark's ability to handle large volumes of data and its in-memory processing capabilities made it an ideal choice for real-time sentiment analysis tasks. Spark was used to clean the data, pre-process it, and apply the sentiment analysis model. Once the data is processed, the resulting sentiment scores are stored in a PostgreSQL database.

Finally, the processed data is sent to the user interface through a WebSocket connection, ensuring real-time updates for the end-users. This modular design gives a clear separation between the different components of the system, allowing for easy troubleshooting and maintenance. This design approach enables future improvements and adaptability, ensuring the system remains relevant and effective as new data sources and processing techniques emerge.

C. Sentiment Analysis Model Selection

Several factors were considered when selecting the most appropriate sentiment analysis algorithm for the project, including model complexity, computational efficiency, scalability, and interpretability. The real-time nature of the application was also considered, which required processing and analysing large amounts of textual data from social media platforms such as Reddit. Various machine learning algorithms commonly used for sentiment analysis, such as logistic regression, support vector machines, and deep learning models, were reviewed initially. However, it was found that some of these algorithms could be too complex and computationally expensive for the real-time application [35].

The Naive Bayes algorithm was identified as a strong candidate due to its simplicity and computational efficiency [6]. The algorithm assumes that each feature, such as a word in the text, is independent of the others, making it more accessible for integration into the data pipeline [12]. The Naive Bayes classifier has been effective in various text classification tasks, including sentiment analysis. Although it may not be the most accurate classifier available, it provides a reasonable trade-off between accuracy and computational complexity. The algorithm is scalable and capable of handling large datasets with many features, making it suitable for processing the vast amount of textual data generated by social media platforms such as Reddit. The model can also be easily updated with new data, allowing it to adapt to changing trends and patterns in sentiment. Finally, the interpretability of the Naive Bayes model, which assigns probabilities to each class based on the observed frequencies of words [12] in the training data, was also considered. This interpretability can help understand the underlying patterns in the data and improve the model if necessary.

After evaluating these factors and considering the specific requirements of the real-time sentiment analysis application, the Naive Bayes algorithm was selected as the most suitable model for the project.

D. Data Storage Component

Considering the potential for scalability and the addition of new tables in the future, the choice to use an SQL based database over Cassandra was carefully considered. While Cassandra is known for its exceptional scalability and high write performance, it sacrifices certain features that are vital for the project's requirements. One of the main strengths of SQL databases is their support for complex querying and data manipulation tasks. As the project grows and new tables are introduced, the ability to perform complex queries, join tables, and create relationships between them will become increasingly important. SQL databases offer these capabilities inherently, while NoSQL databases such as Cassandra have limitations when it comes to handling complex queries and joining multiple tables [7].

Moreover, SQL databases provide strong consistency guarantees [7], which are crucial for maintaining data integrity and ensuring the reliability of the analysis results. Although Cassandra is highly available and can scale horizontally, it uses eventual consistency, which may lead to temporary inconsistencies in the data [23]. This trade-off is not ideal for the project, as the quality of the insights generated depends on the accuracy and consistency of the stored data.

IV. IMPLEMENTATION

A. Building the Sentiment Analysis Model

To build the sentiment analysis model, a dataset of labelled comments and reviews related to the retailers' brands was loaded. This dataset was converted into a parquet file for efficient storage and access [33]. The parquet format was chosen due to its columnar storage, which improves the read and write performance [18], and its ability to compress data efficiently, resulting in reduced storage requirements [29]. Next, stop words are removed from the tokenized text. Stop words are common words such as 'the', 'and' and 'in' that often do not carry any significant meaning or sentiment [15]. Removing these words helps reduce the dimensionality of the data and improve the performance of the model [30]. After the removal of stop words, a bag-of-words representation is created by converting the filtered words into a numerical representation of occurrences of each word in the text. The bag-of-words representation enables the model to quantify the relationship between words and sentiment [30].

Finally, the Inverse Document Frequency (IDF) is calculated for the words in the bag-of-words representation. IDF is a measure that helps to identify the importance of each word by considering its frequency across all documents [16]. The IDF values are then combined with the raw word frequencies to generate a weighted feature vector for each text, which serves as the input to the sentiment analysis model. By applying these pre-processing steps, the data is transformed into a suitable format for training and evaluating the sentiment analysis algorithm. The choice of the appropriate algorithm depends on the dataset and the specific requirements of future tenants. In our context, the Naive Bayes algorithm was used for sentiment analysis due to its simplicity and effectiveness in handling text data [21]. The algorithm was then trained, saved, and evaluated using the Area Under the Curve (AUC) score to assess its performance.

B. Consuming Data from Kafka

The data ingestion pipeline starts with consuming the streaming data from Reddit that has been published to Kafka by the Reddit data extraction component. A Kafka consumer is implemented to subscribe to the appropriate Kafka topic and consume the incoming messages containing the Reddit data.

Apache Spark is employed to process the consumed data efficiently. Spark Streaming, a component of Apache Spark, is specifically designed for processing real-time data streams [9]. It ingests the data from Kafka and processes it in micro-batches, enabling near real-time data processing [9]. A Spark object was created, which is essential for processing the streaming data coming from Kafka. Spark is used to clean the dataset and perform sentiment analysis before storing the results in the PostgreSQL database.

The function built for the above purpose took two inputs: the environment (either production or development) and the application name. Based on the provided environment, it sets the appropriate configuration for the Spark object. If, in a production environment, the Spark object is set to use the 'yarn' cluster manager, while in a development environment, it runs in 'local' mode on the developer's machine. This ensures that the appropriate resources are utilized according to the environment the project is running in. Additionally, the function configures the Spark object with necessary packages and settings, such as the Kafka and PostgreSQL connectors, to allow seamless integration with these external components. It also ensures that the streaming process can be stopped gracefully when required [10]. Upon successful creation of the Spark object, it is returned to be used in the data ingestion pipeline. In case of any errors during the process, appropriate error messages are logged to help identify and resolve the issues. This way, the function ensures that the Spark object is properly configured and ready to handle the streaming data coming from the Reddit comments for sentiment analysis.

C. Data Pre-processing with Spark

After consuming the data from Kafka, the next stage in the pipeline is data pre-processing. The raw text data from Reddit needs to be cleaned and transformed before sentiment analysis can be performed. Apache Spark is used for the pre-processing stage, leveraging its in-memory processing capabilities to efficiently process large volumes of streaming data. The pre-processed data is structured as a 'DataFrame', a distributed collection of data in Spark that allows for optimizations in various data manipulation tasks [2]. Some data cleaning focusing upon removing null values, duplicates and invalid characters was then applied.

D. Storing Results in PostgreSQL

The final component of the data ingestion pipeline is used for storing the processed data, including the pre-processed text data. A PostgreSQL database is used for data storage, offering a robust, scalable, and performant solution for handling large amounts of data.

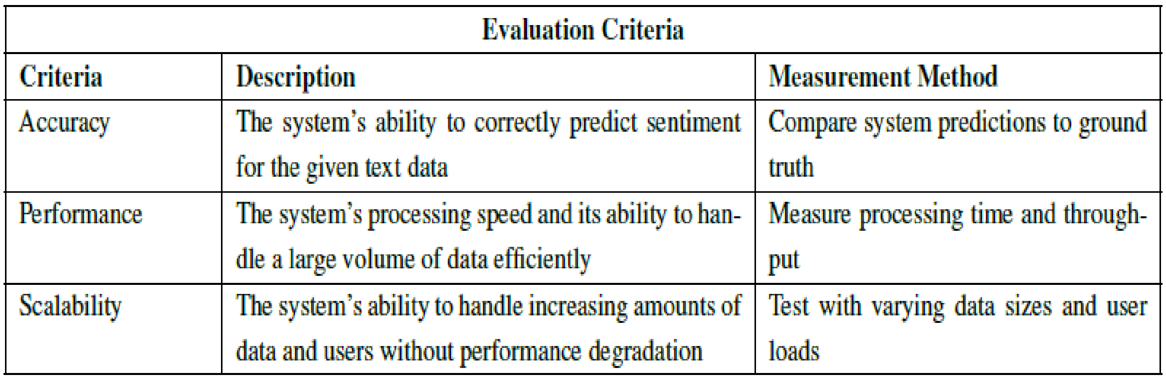
The processed data is stored in a table with a schema that includes columns for the original text, the pre-processed text, and any additional metadata, such as timestamps or post identifiers. This structured storage format enables efficient querying and retrieval of the data for various analytical tasks, such as generating sentiment analytics for the web application.

E. Back-end operations

A job worker component is created where a connection to the data streaming platform (Kafka) is established, and a client for accessing the Reddit platform is created. The system then initiates a stream of comments from a specified subreddit. As new comments arrive, the system processes each one, extracting relevant information such as tenant identification, text, creation time, user, and campaign identifier. The processed comments are then sent to a specific topic on the data streaming platform, where they can be further processed and analysed. The system continuously monitors the progress of the streaming process, updating it every second until the specified duration is reached. Once the duration is reached, the system stops receiving new comments, disconnects from the data streaming platform, and performs clean-up operations to release resources. Throughout the entire process, the system takes care of error handling and ensures smooth execution of the streaming process for the given campaign. A connection is established to handle real-time communication between the server and the web application using a WebSocket. When a new client connects, the system logs the connection event.

V. EVALUATION

A. Evaluation Criteria



Evaluation Criteria		
Criteria	**Description**	**Measurement Method**
Accuracy	The system's ability to correctly predict sentiment for the given text data	Compare system predictions to ground truth
Performance	The system's processing speed and its ability to handle a large volume of data efficiently	Measure processing time and throughput
Scalability	The system's ability to handle increasing amounts of data and users without performance degradation	Test with varying data sizes and user loads

B. Evaluating the sentiment analysis pipeline

To test with a large volume of data, a script was developed to post comments on Reddit using the Reddit API. The dataset used for this script was downloaded from Kaggle, and the reviews were extracted from the dataset. The script posts comments on Reddit using the ‘Snoowrap’ library. Based on this dataset, the script could post 24 comments every two minutes. The script was run until all fake comments were posted. After posting the comments, the latency of the system was calculated by comparing the time the comment was posted to the time the comment was processed by the system. The system maintains a consistent latency, with an average processing time of 1.5 seconds per comment. It is important to note that minor fluctuations in latency occur, ranging between 1.3 to 1.7 seconds. Despite these variations, the system demonstrates efficient performance in handling the incoming comment load. Processing 24 comments every two minutes, the system exhibits the ability to process 720 comments per hour. The consistency and efficiency of the system’s performance in handling a large volume of data within a short time frame validate the scalability and real-time capabilities of the sentiment analysis system.

C. *Evaluation of the sentiment analysis model*

The evaluation of the sentiment analysis model was conducted using a Naive Bayes classifier for real-time analysis. A methodical validation process, including dividing the data into training and test subsets, was used to validate the outcomes. The model achieved an accuracy of 0.6737, successfully classifying roughly 67.37 per cent of the sentiments in the test dataset. This result, while indicating a working model, also suggests room for improvement. An F1 score of 0.7894 signifies a good balance between precision and recall, particularly for positive sentiments, an essential aspect in retail analysis. However, an AUC value of 0.5468, while acceptable, implies a need for further tuning, especially in distinguishing between different sentiment categories. Integration with the Data Version Control (DVC) system enables ongoing refinement to cater to various tenant-specific needs, supporting the model's application across different retail environments. The findings point to future research directions, with the divergence between the F1 score and AUC potentially indicating issues with class balance or feature engineering. Experiments with alternative feature selection, resampling, or model architecture could lead to enhancements. Additional examination of incorrect classifications may also reveal specific areas where the model requires adjustment. These outcomes affirm the viability of using a Naive Bayes classifier for real-time sentiment analysis in online retail and set a foundation for future inquiries. The combination of quantitative measures and adaptability through DVC presents a holistic picture of the model's capabilities and a path for increasing both accuracy and efficiency. This evaluation contributes valuable insights, informing the continuous improvement of a sentiment analysis system tailored to the multifaceted and evolving landscape of multi-tenant online retail.

VI. CONCLUSION

In conclusion, the real-time sentiment analysis system developed in this project effectively addresses the need for efficient and accurate analysis of user-generated content on platforms such as Reddit. The system demonstrates the power of combining data ingestion and sentiment analysis pipelines, leveraging Kafka for streaming, Spark for data processing, and various machine learning models for sentiment prediction. This project not only showcases the potential for real-time sentiment analysis, but also highlights the challenges and opportunities encountered throughout the development process.

A. *Future Work*

Despite the successes achieved in this project, it is crucial to reflect on the limitations and challenges encountered. One significant challenge was selecting the appropriate machine learning model for sentiment prediction. While the Naïve Bayes model performed satisfactorily, deep learning-based algorithms are known to achieve higher accuracy and F1-score at the cost of increased computational expense and training time. This trade-off between accuracy and efficiency is a recurring theme in the field of sentiment analysis, and future work could investigate novel approaches or optimization techniques to address this issue.

Another area for improvement is the scalability and adaptability of the system to various tenants and datasets. The current implementation might require adjustments in the choice of model and tuning parameters depending on the tenant’s dataset. Incorporating a more dynamic and adaptive model selection process could help streamline the onboarding process for new tenants. Furthermore, the project could benefit from a more comprehensive evaluation, incorporating more extensive performance metrics and comparisons with alternative approaches. A more robust evaluation would provide valuable insights into the strengths and weaknesses of the current implementation and inform future improvements.

Lastly, the integration of additional data sources and platforms could significantly enhance the system’s versatility and applicability. Expanding the scope of the project to include other social media platforms, forums, or customer review websites would offer a more comprehensive view of user sentiment and enable more informed decision-making for businesses. In summary, the real-time sentiment analysis system developed in this project showcases the potential of integrating cutting-edge technologies and techniques to deliver valuable insights into user sentiment. Reflecting on the challenges and limitations encountered throughout the project, there are several opportunities for future work, including improving model selection, enhancing scalability and adaptability, conducting more comprehensive evaluations, and expanding data source integrations. By addressing these areas, the system could continue to evolve and provide even more significant value to businesses seeking to understand and respond to the sentiments expressed by their customers in real time.

REFERENCES

[1] Al-Otaibi, S. T., Alnassar, A., Alshahrani, A., Al-Mubarak, A., Albugami, S., Almutiri, N., & Albugami, A. "Customer Satisfaction Measurement using Sentiment Analysis". International Journal of Advanced Computer Science and Applications, *9*, 2018.

[2] Armbrust, M., Das, T., Torres, J., Yavuz, B., Zhu, S., Xin, R., Ghodsi, A., Stoica, I., & Zaharia, M. A. "Structured Streaming: A Declarative API for Real-Time Applications in Apache Spark". Proceedings of the 2018 International Conference on Management of Data, 2018.

[3] Atteveldt, W. van, Velden, M. A. C. G. van der, & Boukes, M. "The Validity of Sentiment Analysis: Comparing Manual Annotation, Crowd-Coding, Dictionary Approaches, and Machine Learning Algorithms". Communication Methods and Measures, *15*, 121–140, 2021.

[4] Aung, T., & Min, H. Y. "Evaluation of Apache Kafka in Real-Time Big Data Pipeline Architecture",2018.

[5] Baltas, A., Kanavos, A., & Tsakalidis, A. K. "An Apache Spark Implementation for Sentiment Analysis on Twitter Data". International Workshop on Algorithmic Aspects of Cloud Computing, 2016.

[6] Berrar, D. P. " Bayes' Theorem and Naive Bayes Classifier." Encyclopedia of Bioinformatics and Computational Biology,2019.

[7] Binani, S., Gutti, A., & Upadhyay, S. "SQL vs. NoSQL vs. NewSQL- A Comparative Study." Communications on Applied Electronics, *6*(1), 43–46. 2016. https://doi.org/10.5120/cae2016652418

[8] Cambria, E., Schuller, B., Liu, B., Wang, H., & Havasi, C. "Knowledge-Based Approaches to Concept-Level Sentiment Analysis", IEEE Intelligent Systems, *28*(2), 12–14.2013. https://doi.org/10.1109/MIS.2013.45

[9] Chintapalli, S., Dagit, D., Evans, B., Farivar, R., Graves, T., Holderbaugh, M., Liu, Z., Nusbaum, K., Patil, K., Peng, B. J., & Poulosky, P. "Benchmarking Streaming Computation Engines: Storm, Flink and Spark Streaming". 1789–1792,2016. https://doi.org/10.1109/IPDPSW.2016.138

[10] Drohobytskiy, Y., Brevus, V., & Skorenkyy, Y. "Spark Structured Streaming: Customizing Kafka Stream Processing". 2020 IEEE Third International Conference on Data Stream Mining & Processing (DSMP), 296–299, 2020. https://doi.org/10.1109/DSMP47368.2020.9204304

[11] Fahmi, M. A., Hidayat, S., & Hidayatullah, A. F. " Application of Lexicon Based for Sentiment Analysis of COVID-19 Booster Vaccinations on Twitter Social Media Using Naïve Bayes Method". Jurnal Teknik Informatika (Jutif), 2022.

[12] Fathima, S., & Hundewale, N. "Comparison of classification techniques-SVM and naives bayes to predict the Arboviral disease-Dengue". 538–539, 2011. https://doi.org/10.1109/BIBMW.2011.6112426

[13] Gopani, A. "How Uber is Leveraging Apache Kafka For More Than 300 Micro Services. Analytics India Magazine". https://analyticsindiamag.com/how-uber-is-leveraging-apache-kafka-for-more-than-300-micro-services/, 2021.

[14] Hartmann, J., Heitmann, M., Siebert, C., & Schamp, C. "More than a Feeling: Accuracy and Application of Sentiment Analysis". International Journal of Research in Marketing, *40*(1), 75–87, 2023. https://doi.org/10.1016/j.ijresmar.2022.05.005

[15] Hickman, L., Thapa, S., Tay, L., Cao, M., & Srinivasan, P. "Text Preprocessing for Text Mining in Organizational Research: Review and Recommendations". Organizational Research Methods, *25*, 114–146, 2020.

[16] Hu, K., Wu, H., Qi, K., Yu, J., Yang, S., Yu, T., Zheng, J., & Liu, B. "A domain keyword analysis approach extending Term Frequency-Keyword Active Index with Google Word2Vec model". Scientometrics, *114*, 1031–1068, 2018.

[17] Hutto, C., & Gilbert, E. "VADER: A Parsimonious Rule-Based Model for Sentiment Analysis of Social Media Text." Proceedings of the International AAAI Conference on Web and Social Media, *8*(1), Article 1,2014. https://doi.org/10.1609/icwsm.v8i1.14550

[18] Ivanov, T., & Pergolesi, M. "The impact of columnar file formats on SQL-on-hadoop engine performance: A study on ORC and Parquet". Concurrency and Computation: Practice and Experience, *32*(5). https://doi.org/10.1002/cpe.5523, 2020

[19] Kim, Y. "Convolutional Neural Networks for Sentence Classification". Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 1746–1751. https://doi.org/10.3115/v1/D14-1181, 2014.

[20] Kim, Y.-K., & Jeong, C.-S. Large Scale Image Processing in Real-Time Environments with Kafka,2017.

[21] Kowsari, K., Meimandi, K. J., Heidarysafa, M., Mendu, S., Barnes, L. E., & Brown, D. E. "Text Classification Algorithms: A Survey". Information, *10*(4), 150, 2019.https://doi.org/10.3390/info10040150

[22] Kul, S., Tashiev, I., Sentas, A., & Sayar, A. "Event-Based Microservices With Apache Kafka Streams: A Real-Time Vehicle Detection System Based on Type, Color, and Speed Attributes". IEEE Access, *9*, 83137–83148, 2021.

[23] Lakshman, A., & Malik, P. "Cassandra: A decentralized structured storage system". ACM SIGOPS Oper. Syst. Rev., 44, 35–40, 2010.

[24] Madni, G. R. "Consumer's Behavior and Effectiveness of Social Media". Global Journal of Management and Business Research, 14, 2015.

[25] Malviya, S., Tiwari, A. K., Srivastava, R., & Tiwari, V. "Machine Learning Techniques for Sentiment Analysis: A Review". SAMRIDDHI : A Journal of Physical Sciences, Engineering and Technology, *12*, 72–78, 2020.

[26] Nasteski, V. "An overview of the supervised machine learning methods". Horizons, 4, 51–62, 2017.

[27] Nodarakis, N., Sioutas, S., Tsakalidis, A. K., & Tzimas, G." Using Hadoop for Large Scale Analysis on Twitter: A Technical Report". ArXiv, abs/1602.01248, 2016.

[28] Palit, S., & Ghosh, S. "Real Time Sentiment Analysis", Int. J. Synth. Emot., *11*, 27–35, 2020.

[29] Plase, D., Niedrite, L., & Taranovs, R. "A Comparison of HDFS Compact Data Formats: Avro Versus Parquet". Mokslas - Lietuvos Ateitis, *9*, 267–276, 2017.

[30] Pradana, A. W., & Hayaty, M. "The Effect of Stemming and Removal of Stopwords on the Accuracy of Sentiment Analysis on Indonesian-language Texts". Kinetik: Game Technology, Information System, Computer Network, Computing, Electronics, and Control, 2019.

[31] Prakash, Y. "The DVC Guide: Data Version Control For All Your Data Science Projects". Medium, https://towardsdatascience.com/the-dvc-guide-data-version-control-for-all-your-data-science-projects-382d5b5aab00, 2023.

[32] Rodrigues, A. P., Rao, A. S., & Chiplunkar, N. N. "Sentiment Analysis of Real Time Twitter Data Using Big Data Approach". 2017 2nd International Conference on Computational Systems and Information Technology for Sustainable Solution (CSITSS), 1–6, 2017.

[33] Saeedan, M., & Eldawy, A. "Spatial parquet: A column file format for geospatial data lakes". Proceedings of the 30th International Conference on Advances in Geographic Information Systems, 2022.

[34] Usha, V., & Prema, K. "Aspect Based Sentiment Analysis Using Rule Based Approach", 2021 First International Conference on Advances in Computing and Future Communication Technologies (ICACFCT), 48–51, 2021.

[35] Wankhade, M., Rao, A. C. S., & Kulkarni, C. "A survey on sentiment analysis methods, applications, and challenges". Artificial Intelligence Review, 55, 5731–5780, 2022.

[36] Zaharia, M. A., Xin, R., Wendell, P., Das, T., Armbrust, M., Dave, A., Meng, X., Rosen, J., Venkataraman, S., Franklin, M. J., Ghodsi, A., Gonzalez, J. E., Shenker, S., & Stoica, I. "Apache Spark: A unified engine for big data processing". Commun. ACM, *59*, 56–65, 2016.