Brand Recognition and Sentiment Analysis from Online Comments in YouTube

Ly Tuan Khoa[,†], Tran Trung Kien[,†], Nguyen Toan Tien Cuong[,†],Pham Kim Long[,†], Do Trong Hop[,†]

Faculty of Information Science and Engineering, University of Information Technology
[†]Vietnam National University, Ho Chi Minh City, Vietnam.

Abstract—In the digital age, brand perception is heavily influenced by online comments and reviews. With the increasing popularity of platforms like YouTube, it is crucial for companies to understand and analyze public sentiment towards their brands. This study proposes a hybrid system that combines brand recognition with sentiment analysis to assess consumer opinions from YouTube comments. By leveraging advanced natural language processing techniques and deep learning models, we predict sentiment and identify brand mentions with high accuracy. To ensure scalability and efficiency, both brand recognition and sentiment analysis models are trained and deployed using Apache Spark, SpaCy, and Textblob which are frameworks for big data processing and distributed deep learning training. This research highlights the importance of integrating real-time sentiment analysis with brand monitoring to enhance marketing strategies and customer engagement.

Index Terms—Brand Analysis, Sentiment Analysis, Online Reviews, YouTube Data, Deep Learning, Big Data

Introduction

With YouTube's vast reach and influence, it has become a crucial platform for brands to engage with their audience and assess public sentiment. Every year, numerous brands invest heavily in creating content for YouTube, aiming to capture consumer interest and loyalty. However, understanding the impact of these efforts requires meticulous analysis of viewer comments, which can span millions of interactions across videos.The study deploys datasets extracted from YouTube using its Data API.[1] According to the obtained results, high filtering accuracy (more than 98 percent) can be achieved with low-complexity algorithms, implying the possibility of developing a suitable browser extension to alleviate comments for analyzing comments on YouTube in future.

Planning marketing strategies or evaluating campaign effectiveness always requires careful consideration of audience engagement, sentiment, and brand recognition. While some aspects of audience reactions are predictable, comments can vary widely, making it challenging to extract meaningful insights. This process can be time-consuming and resource-intensive, potentially leading to missed opportunities for timely responses or adjustments to marketing strategies.

Therefore, it is ideal to be able to analyze and predict brand sentiment and recognition trends efficiently. Such decisions can be based on advanced sentiment analysis and brand recognition models that process large volumes of comment data. This approach not only helps in understanding current audience sentiment but also in forecasting future trends.

The contribution of this paper is two-fold. First, we applied spaCy and other relative libraries for identifying key areas of focus in brand sentiment and recognition based on YouTube comments. Second, we

applied Spark to identifying sentiments and brand mentions over time. The input data used for these models are extensive comment datasets from various brand-related videos and metadata provided by YouTube. Given the large volume of data collected over years, traditional machine learning libraries may not suffice for training these models. To ensure the system's practicality and efficiency, all recommendation and sentiment analysis models are trained and deployed using big data technology. Specifically, identifying models are trained using Spark MLlib, while the deep learning-based sentiment analysis models are trained in a distributed manner using the BigDL library. All models are deployed on Spark,SpaCy and textBlob enabling the system to scale and process vast amounts of data required for effective brand sentiment analysis and recognition.

Related Works

There have been numerous studies developing systems for various topics involving large amounts of data on a daily basis, such as movie recommendations using the MovieLens dataset, orthe Netflix platforms. Different techniques and methods have been employed to build these systems, including content-based filtering, collaborative filtering, and hybrid recommenders using weighted combination techniques. However, these approaches are not directly applicable to the task of analyzing brand recognition and sentiment from online comments on YouTube.

Other approaches exist that analyze the spatiotemporal distribution of sentiment in relation to various influencing factors. Many of these studies focus on understanding sentiment trends at a given point in time. The methods usually consist of two main blocks: analyzing sentiment through natural language processing (NLP) techniques on large datasets and then interpreting this analysis using different types of machine learning models.

This study focuses on recognizing company brands and analyzing positive and negative comments from YouTube data using Spark Apache. We employ NLP techniques to process and analyze large volumes of comment data, extracting brand recognition and sentiment insights. By utilizing various machine learning models within the Spark MLlib framework,spacy, and TextBlob, we can efficiently handle large datasets and comprehensively analyze public perception towards different brands.

Dataset

Catch Notes Dataset

The core dataset used in this project consists of comments collected via the YouTube API[1]. The dataset includes comments from videos spanning several years, offering insights into public sentiment and brand recognition. Each comment entry includes information such as the text of the comment, the user who posted it, the time it was posted, and other metadata. For instance, the API call retrieves details like when the comment was posted, the user's channel information, and likes or replies to the comment.

The dataset is substantial, given the high volume of user interactions on YouTube videos. Features of interest include the text content of comments, the number of likes or replies, and timestamps. This rich dataset allows for comprehensive analysis of sentiment and brand mentions over time. Depending on the requirements of each experiment, the dataset can be filtered and reduced in size, focusing on specific keywords or time periods.

In this project, we utilized a YouTube API key to fetch comments from specified videos. The process involved creating a YouTube service object, specifying the video ID, and retrieving up to 10000 comments, collected from users' comments on YOUTUBE videos. Each comment is assigned one of 3 sentiment: Positive (1), Neutral (0), and Negative (-1) sentiment in the Fig[1]. The analysis involves collecting and statistically analyzing comments behaviours among mentioning brand names from YouTube videos in fig2.

[1]https://developers.google.com/youtube/v3/docs/commentThreads/list

No.	Comments	Brand	Labels
1	Samsung good	Samsung	1
2	Apple Sucks	Apple	-1
3	Korean Samsung	Samsung	0

Figure 1: Data Label

Proposed Methods

System architecture

The diagram Fig[1] depicts a real-time data processing pipeline for YouTube comments using Apache Spark . YouTube serves as the data source, providing comments that are collected and ingested into Spark dataframe. The data, where machine learning models are trained and applied to live data. Within Apache Spark, SpaCy, and TextBlob, the data is processed using Data Frames and queried with Spark SQL. The processed results are stored in a cloud service and visualized on a dashboard, offering real-time insights into brand recognition and sentiment analysis. This pipeline leverages big data technologies to efficiently manage and analyze large volumes of data, providing actionable marketing insights.

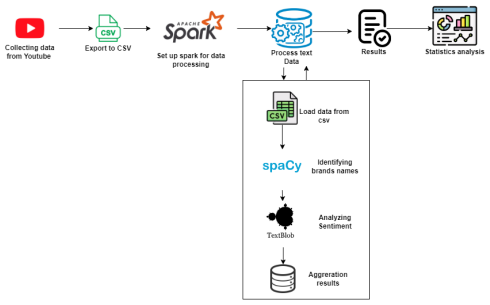


Figure 2: System

Apache Spark [3]Apache Spark is a consolidated big data analytics engine and provides absolute data parallelism. This paper scrutinizes a technical review on big data analytics using Apache Spark and how it uses in-memory computation that makes it remarkably faster as compared to other corresponding frameworks.

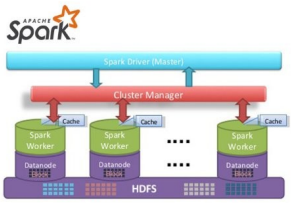


Figure 3: Spark apache architechture

Spark DataFrame A Spark DataFrame is an integrated data structure with an easy-to-use API for simplifying distributed big data processing. DataFrame is available for general-purpose programming languages such as Python, and Scala.

It is an extension of the Spark RDD API optimized for writing code more efficiently while remaining powerful. [4]

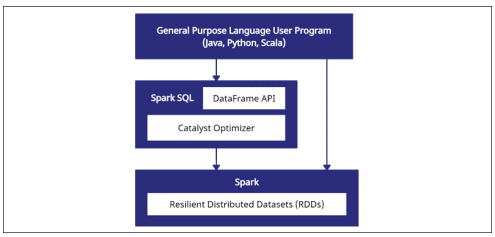


Figure 4: Spark DataFrame

SpaCy SpaCy is a free, open-source library for advanced Natural Language Processing (NLP) in Python. If you're working with a lot of text, you'll eventually want to know more about it. For example, what's it about? What do the words mean in context? Who is doing what to whom? What companies and products are mentioned? Which texts are similar to each other? spaCy is designed specifically for production use and helps you build applications that process and "understand" large volumes of text. It can be used to build information extraction or natural language understanding systems, or to pre-process text for deep learning. We build our dataset by scraping YouTube

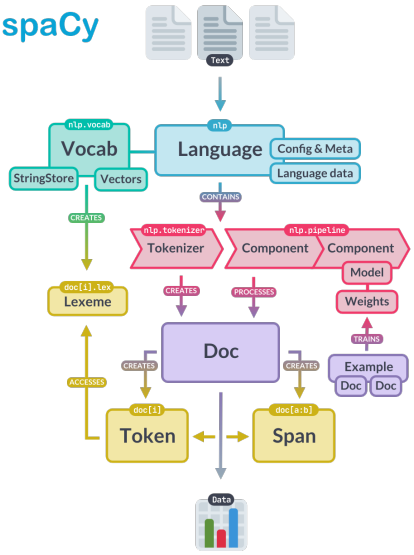


Figure 5: SpaCy

comments. We use the YouTube Data API for authenticating and accessing the comments on videos [20]. First, the API is used for authentication and then the credentials obtained are fed into the comment extractor. The comment extractor then scrapes comments from the comment section by scrolling through all the comments and loading them dynamically. Fig.6 shows the scraping process The dataset consists of 10,000 comments picked from different tutorial videos. We chose brand value history videos for our experiments because they contain a wide variety of comments. Positive tells that the viewers perceived the content as worthy and that the content created a positive impact on them. Negative provides information on what is wrong with the content and why the viewers are not attracted to it.

Data Collection

The YouTube Data API allows developers to access and interact with YouTube content programmatically, including retrieving comments from YouTube videos. To use the API, start by creating a project in the Google Developers Console, enable the YouTube Data API v3, and obtain an API key. With the API key, construct a request URL to fetch comments from a specific video by its unique identifier (video ID). The basic request URL format is https://www.googleapis.com/youtube/v3/commentThreads?part=snippet&videoId=VIDEO_ID&key=YOUR_API_KEY, where `VIDEO_ID` is the actual identifier of the video and `YOUR_API_KEY` is the API key. The API returns results in pages, so it is necessary to handle pagination by using the `nextPageToken` from the response to fetch subsequent pages of comments. The API call retrieves the comments in JSON format, which can then be processed and analyzed for various purposes such as sentiment analysis or keyword extraction. Handling pagination ensures that all comments are retrieved and available for analysis.

Figure 6: Data Collection

Data Pre-processing

Preprocessing is a crucial step in preparing text data for further analysis. In this notebook, preprocessing involves converting the text to lowercase, removing non-alphanumeric characters, and splitting the text into individual words. These steps help normalize the text data, making it easier to analyze and ensuring consistency. Lowercasing ensures that words

are treated uniformly regardless of their original case, while removing non-alphanumeric characters eliminates punctuation and special symbols that might not be relevant for the analysis. Splitting the text into words allows for tokenization, which is essential for many natural language processing tasks such as brand recognition and sentiment analysis.

Brand Recognition

The code and the resulting DataFrame output show a process where comments have been preprocessed into words and checked for the presence of brand names, with the results displayed in a tabular format. The vast amount of text data contains a huge amount of information. An important aspect of analyzing these text data is the identification of Named Entities.

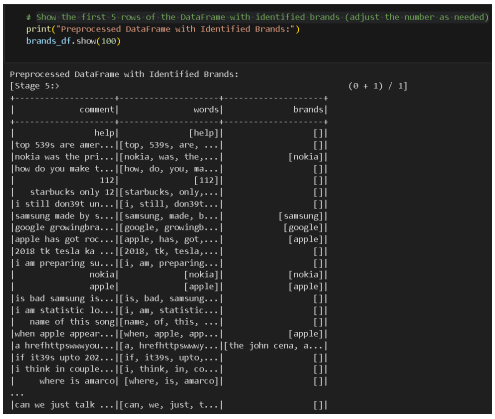


Figure 7: Brand recognition

Analyzes the sentiment of a comment using TextBlob

TextBlob's sentiment analysis works by using a trained machine learning model to classify the sentiment of a given text. It considers the words and their arrangement to assign a polarity (positive, negative, or neutral) and subjectivity score to the text. TextBlob is a Python library that simplifies text processing, including tasks like part-of-speech tagging, noun phrase extraction, and sentiment analysis. It provides a simple API for diving into common natural language processing tasks. TextBlob uses a Naive Bayes classifier for sentiment analysis. It is trained on a labeled dataset containing examples of text with associated sentiment labels (positive, negative, or neutral). The sentiment score in TextBlob's polarity ranges from -1 to 1, where -1 represents a highly negative sentiment, 0 is neutral, and 1 indicates a highly positive sentiment. The subjectivity score ranges from 0 to 1, with 0 being objective and 1 being subjective. [2]

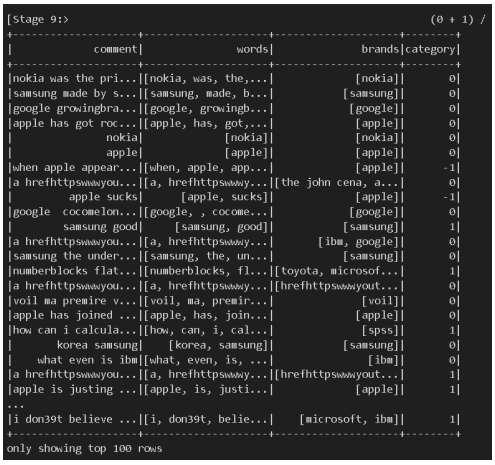


Figure 8: Few rows to see the sentiment categories

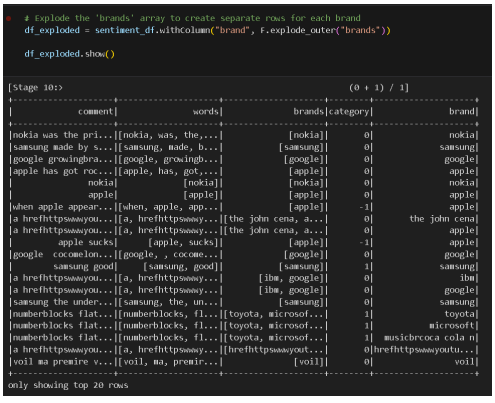
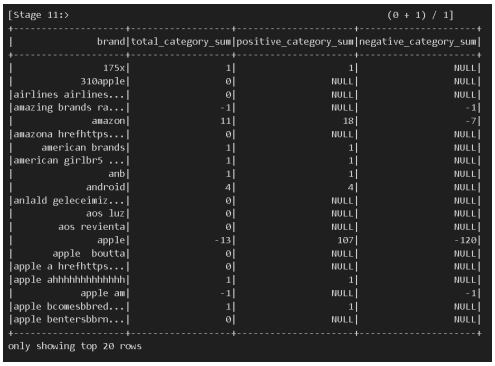


Figure 9: Sentiment analysis

Calculate total category sum,positive category sum, negative category sum

```
# Calculate total category sum for each brand
total_category_sum_by_brand = df_exploded.select("brand", "category").groupby("brand").agg(
    f.sum("category").alias("total_category_sum")
)

# Calculate positive category sum for each brand (filter and sum)
positive_category_sum_by_brand = df_exploded.select("brand", "category").where(f.col("category") > 0).groupby("brand").agg(
    f.sum("category").alias("positive_category_sum")
)

# Handle brands with no positive categories by filling with 0
positive_category_sum_by_brand = positive_category_sum_by_brand.withColumn("positive_category_sum", f.coalesce(f.col("positive_category_sum"), f.lit(0)))

# Calculate negative category sum for each brand (filter and sum)
negative_category_sum_by_brand = df_exploded.select("brand", "category").where(f.col("category") < 0).groupby("brand").agg(
    f.sum("category").alias("negative_category_sum")
)

# Handle brands with no negative categories by filling with 0
negative_category_sum_by_brand = negative_category_sum_by_brand.withColumn("negative_category_sum", f.coalesce(f.col("negative_category_sum"), f.lit(0)))

# Join the three dataframes to get all columns for each brand
brand_category_sum = total_category_sum_by_brand.join(positive_category_sum_by_brand, on="brand", how="outer").join(negative_category_sum_by_brand, on="brand", how="outer")

# Display the result
brand_category_sum.show()
```

Figure 10: Code to calculate total sum in each situations



```
[Stage 11:>                                                         (0 + 1) / 1]
+--------------------+--------------------+--------------------+--------------------+
|               brand|total_category_sum|positive_category_sum|negative_category_sum|
+--------------------+--------------------+--------------------+--------------------+
|                175x|                   1|                   1|                NULL|
|           310apple|                   0|                NULL|                NULL|
|airlines airlines...|                   0|                NULL|                NULL|
|amazing brands ra...|                  -1|                NULL|                  -1|
|              amazon|                  11|                  18|                  -7|
|amazona hrefhttps...|                   0|                NULL|                NULL|
|     american brands|                   1|                   1|                NULL|
|american girlbr5 ...|                   1|                   1|                NULL|
|                anb|                   1|                   1|                NULL|
|             android|                   4|                   4|                NULL|
|anlald geleceimiz...|                   0|                NULL|                NULL|
|            aos luz|                   0|                NULL|                NULL|
|        aos revienta|                   0|                NULL|                NULL|
|               apple|                 -13|                 107|                -120|
|       apple  boutta|                   0|                NULL|                NULL|
|apple a hrefhttps...|                   0|                NULL|                NULL|
|apple ahhhhhhhhhhhhh|                   1|                   1|                NULL|
|           apple am|                  -1|                NULL|                  -1|
|apple bcomesbbred...|                   1|                   1|                NULL|
|apple bentersbbrn...|                   0|                NULL|                NULL|
+--------------------+--------------------+--------------------+--------------------+
only showing top 20 rows
```

Figure 11: Total category sum,positive category sum, negative category sum

Calculates and displays the total category sum, positive category sum, and negative category sum for each brand in a DataFrame that has been "exploded" (i.e., values in columns containing lists have been split into separate rows) Calculate total category sum for each brand: The first part of the code calculates the total category sum for each brand by grouping the data by "brand" and then summing the "category" column for each group. Calculate positive category sum for each brand: The second part of the code calculates the positive category sum for each brand. It first filters the data to only include rows where the "category" value is greater than 0, then groups the data by "brand" and sums the "category" column. Handle brands with no positive categories: Since some brands may not have any positive categories, the code uses the withColumn and coalesce functions to fill the "positive category sum" column with 0 for those brands. Calculate negative category sum for each brand: Similar to the positive category sum, the third part of the code calculates the negative category sum for each brand. It filters the data to only include rows where the "category" value is less than 0, then groups the data by "brand" and sums the "category" column. Handle brands with no negative categories: As with the positive category sum, the code uses withColumn and coalesce to fill the "negative category sum" column with 0 for brands that do not have any negative categories. Join the three DataFrames: Finally, the code joins the three DataFrames (total category sum, positive category sum, and negative category sum) on the "brand" column to create a single DataFrame with all the calculated values for each brand. Display the result: The last line of the code displays the final DataFrame using the show() method.

Experiment Results

brand	category_sum
samsung	73
google	54
microsoft	48
toyota	25
amazon	11
pepsi	10
intel	8
disney	7
sony	5
android	4
coke	4

Figure 12: Category sum

Category sum Top Brands: Samsung, Google, and Microsoft are the top brands in terms of recognition or positive sentiment. Middle Tier: Toyota has moderate recognition or sentiment. Lower Tier: Brands like Amazon, Pepsi, Intel, Disney, Sony, Android, and Coke have relatively lower scores, indicating less recognition or positive sentiment. If this data is related to brand recognition or sentiment, the scores likely reflect how well-known these brands are or how positively they are viewed in a specific category or region.

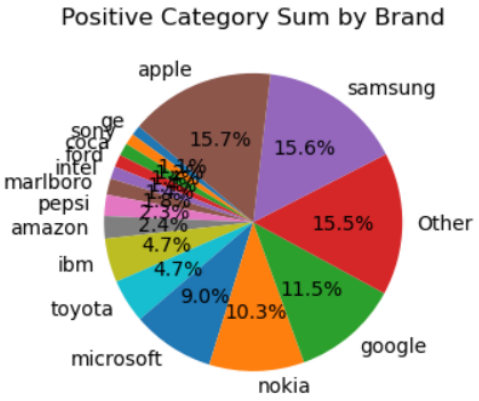


Figure 13: The percentage of positive comments about companies' brands

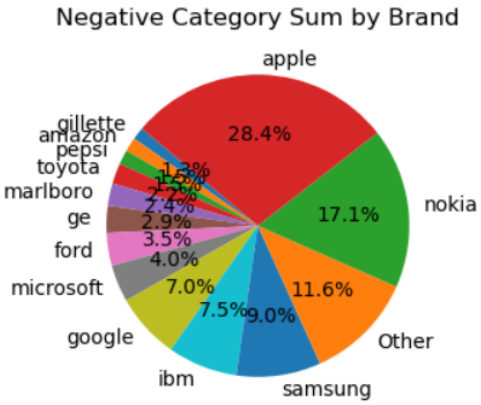


Figure 14: The percentage of negative comments about companies' brands

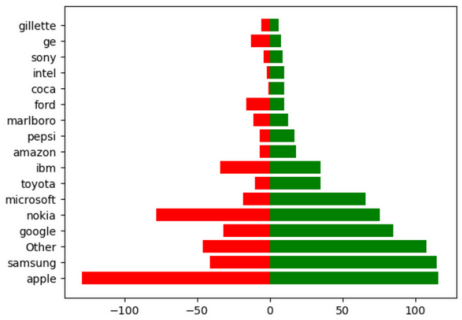


Figure 15: Diverging bar chart of positive and negative values for different categories

Conclusion

Understanding brand recognition and sentiment from online comments is essential for effective marketing strategies. Accurate analysis of consumer feedback can lead to more efficient use of resources, reduced operational costs, and ultimately enhance the profitability of brands. This study proposes approaches using big data frameworks and applications to solve practical problems in brand recognition and sentiment analysis. We have developed deep learning-based multi-variate time series models to analyze and forecast sentiment trends and brand mentions from YouTube comments. To ensure the practicality of the application, all models are trained and deployed using big data technologies. The results show that the proposed models achieve high performance in analyzing brand sentiment and predicting trends.

Future development

Collect and Aggregate Data Continuously scrape and collect comments from your brand's YouTube videos, as well as competitor videos. Organize the data into a structured format, such as a database or spreadsheet, including details like username, comment text, video title, timestamp, etc.

Perform Sentiment Analysis Utilize natural language processing (NLP) techniques to analyze the sentiment expressed in the comments. Implement machine learning models, such as BERT or RoBERTa, that are pre-trained on large text corpora and fine-tuned on your brand's data. Categorize comments as positive, negative, or neutral based on the sentiment scores.

Continuously Iterate and Improve Monitor the performance of your sentiment analysis and brand management efforts over time. Refine your models, data collection processes, and engagement strategies based on the evolving needs and preferences of your audience. Stay up-to-date with the latest advancements in NLP and sentiment analysis to ensure your approach remains effective.

Acknowledgment

We extend our heartfelt gratitude to Dr. Do Trong Hop for your invaluable contributions to our learning journey. Your insightful lectures have not only imparted profound knowledge but have also ignited a sense of enthusiasm and passion within our team, serving as a guiding force that propelled us towards the successful completion of our project. We are deeply appreciative of your unwavering dedication and mentorship.

In expressing our sincere appreciation to Dr. Do Trong Hop, we would like to highlight our profound gratitude for the trust and permission you graciously granted us to carry out our project. This opportunity has been instrumental in our personal and professional growth, allowing us to apply our learning in a real-world context and fostering invaluable skills.

Last but not least, we express our deep thanks to Dr. Do Trong Hop for your tireless commitment. Your steadfast guidance, encouragement, and motivation have been pivotal in steering us through challenges and fostering an environment of continuous improvement and innovation. Dr. Do Trong Hop's mentorship has not only enriched our project experience but has also left an indelible mark on our personal and professional lives.

References

[1] Abdullah O. Abdullah; Mashhood A. Ali; Murat Karabatak; Abdulkadir Sengur,"A comparative analysis of common YouTube comment spam filtering techniques" March 2018, [online] Available:https://ieeexplore.ieee.org/document/8355315

[2] Mohit Kumar Barai,"Sentiment Analysis with TextBlob and Vader" 21 February 2018,[online] Available:https://www.analyticsvidhya.com/blog/2021/10/sentiment-analysis-with-textblob-and-vader/

[3] Eman Shaikh; Iman Mohiuddin; Yasmeen Alufaisan; Irum Nahvi "Apache Spark: A Big Data Processing Engine" 10 February 2020, [online] Availeble:https://ieeexplore.ieee.org/abstract/document/8988541

[4] Milica Dancuk:"What Is a Spark DataFrame?" [online], Availeble:https://phoenixnap.com/kb/spark-dataframe