

**Author: Hendrik A. Dreyer**

**Student ID: 13622464**

**Course: Master of Data Science**

**Faculty: School of Science, Engineering and IT**

# ****Subject: MA5851 – Data Science Master Class 1****

**Assessment: 4 – Strategic Insights Report**

**Due Date: 9 December 2019**

# An investigation into the correlation between

# media and general populace sentiment

# 

## **Executive summary**

Clearly articulate to the reader an overview of the insight’s reports.

## Introduction

My mother used to tell me, “Son, first impressions last. So, pull up your socks, comb your hair and smile, dammit!” We all know that’s true. We also know that the media knows it and they use it like no one else. The effect of the media on the general populace has been well documented and it is now an established fact that the media steer general opinion, alter perception and influence sentiment (Tan, Phang, Chin, Alfred, & Anthony, 2014). Therefore, the art of writing a good headline has become tantamount to a good opening move in a chess game. Headlines serve the purpose of framing the text that follows. It tells you what kind of article you are about to read and by drawing attention to certain details or facts it activates certain memories in your brain (Surber & Schroeder, 2007).

Why is this important? Why should we be bothered by it? In today’s frantic lifestyle attention spans are super short and energy levels are rock-bottom low. This is in many cases due to over exertion, over extension and the obsession to overachieve due to mostly dumb people telling us through stupid entities such as self-help books, videos and phone apps that we can be the next billionaire if we just try harder and believe more. For those reasons, headlines need to be able to grab your attention in the shortest possible timeframe and sucker you into reading the rest of the article (Konnikova, 2014).

This document reports on work that investigated a potential phenomenon that is related to the effect that headlines have on the general populace. Although the relationship between headlines and article content has been documented, not much have been published or spoken about the effect that headlines have on user comments. One might argue that comments posted by users based on articles read is in any case influenced by the content of the articles itself and therefore it is the content of the article that influences user comment and not the title. This investigation turns the question on its head and rather asks, “What are the effects of headlines on user comments?” *The answer(s) to this question can feed directly into numerous industries’ thought devices which in turn can serve as inputs to strategic decision making. For instance, if we know for sure that headlines have a certain effect on user sentiment, then much needed strategic adjustments can be initiated on the back of news articles that are released. In other words, business might find value the anticipated sentiment movement of user and capitalise on it by ways of well placed advertisements.*

Seeing that headlines does the initial framing of the context around the article; it sets the expectation of the user and then releases the content onto the reader. This is an important point as the title now acts as an initiator before the user starts to read the article (Surber & Schroeder, 2007). In order to address the question, “What is the effect of headlines on user comments?”, we need to have access to a large number of user generated comments, that are generated on the back of numerous and various articles that were read, which spans thousands of websites. Individual news sources and article sources are maintained by single entities such as The New York Times, Tech Crunch, etc. the comments generated on those sights might have a certain variety to itself, but it is also confounded to the user, subscriber or reader. In order to find a diverse source that amalgamates several thousand different sources’ user comments, we need to look no further than the forum called, [Hacker News](https://news.ycombinator.com/).

## Data collection

Hacker News, (<https://news.ycombinator.com/>), is a social news website focusing on computer science and entrepreneurship. It is run by Paul Graham's investment fund and startup incubator, Y Combinator. In general, content that can be submitted is defined as "anything that gratifies one's intellectual curiosity (“Hacker News,” 2019).

HN is a public website whereby people mostly post links to media articles. Users can then comment on the posted links and thus public dialogue is encouraged. HN devised a complex scoring algorithm whereby posts are assigned points. However, it is important to mention that the posts are scored and not the comments, although the number of comments on a post contributes towards the score of the post. The other important point to remember about the posts is that the actual wording of a post is not (probably 99% of the time) the words posted by the user, it is in fact the words directly related to the title of the article that the post refers to. In other words, it is the written word as published by the media. Therefore, any corpus derived from the scraped titles are content generated by the media. Herein lies an interesting phenomenon as we now can look into the narrative that the media shapes around all things contemporary.

Looking at the top scored posts, the industry related question(s) that arises from this information revolves around the question of why the media would word headlines the way they do? And, equally important, firstly, why are users of the forum posting these specific articles, and secondly, why are users responding more to certain articles than others. By understanding the forces that push and pulls users to react in certain ways to posted links on the forum can be utilised as an effective insight into various industry activities such as, marketing, opinion polls, voting, etc.

## Content Layout

The site, <https://news.ycombinator.com/>, has a simple layout as far as content is concerned. All posts, dating back to the initial start-up date of the forum, can be accessed via a top-level menu option eloquently labelled, **past**. It is through this option that the author managed to harvest the story headlines for the top 30 stories of each day in the month of August in 2019.

*Note: Assessment 3 harvested the top 30 post per day for the last 365 days. But, due to time, memory and processing power constraints the author decided to lessen the data footprint to only a month’s worth of deadlines.* Each day’s front page contains the top 30 scored posts for the day. Thus, the scraped data represents the highest scored posts per day for each of the 31 days in August 2019.

Assessment 3 further discusses and reports on the webpage layout of Hacker News. The important point here is that only headlines for August 2019 will be used further in this investigation. The second part of the data needed for the investigation resides in the actual comments, which are associated with each of the posted stories. There are several ways to obtain this data. The one way would be to scrape the comments in a similar manner as the author scraped the story headlines. Unfortunately, this involves a much more complex scraping program than was implemented for the headline scraper program. Therefore, the author opted to obtain the user comments via Google BigQuery.

Hacker News decided in March 2016 to publish all stories and comments from Hacker News to the public data section of Google’s BigQuery. One might argue that both the stories and comments can be obtained via Google’s BigQuery and that the scraping was unnecessary. That is besides the point! The point is that there is more than one way to obtain data. An alternative would be to obtain the data via an exposed API, if Hacker News decided to expose one for data consumption. In any case, Figure 1 below depicts the SQL statement as developed by the author in Google BigQuery, which was responsible for harvesting all of the comments associated to each of the top 30 stories of each day in August 2019.

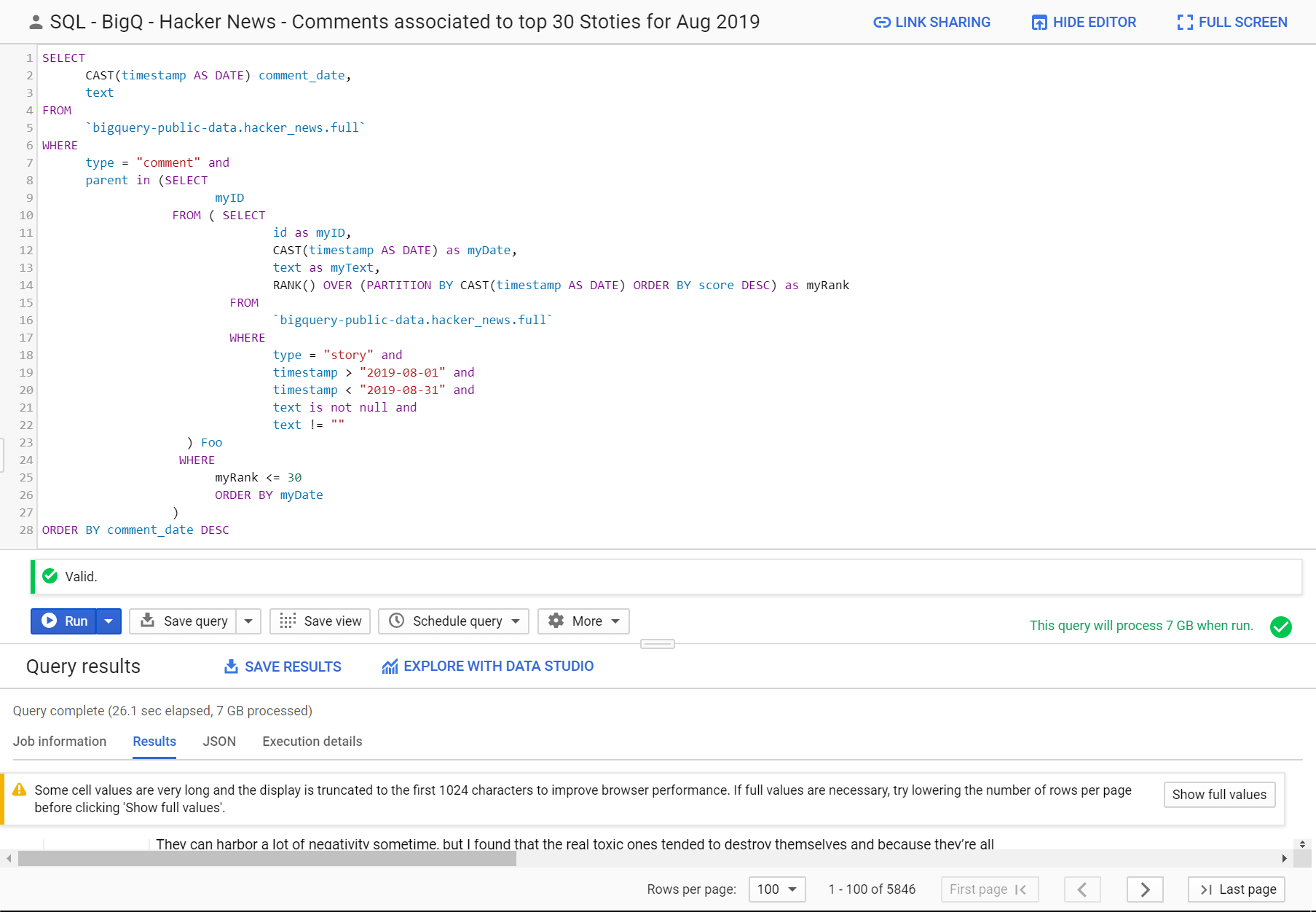


Figure 1 - Google BigQuery SQL for all comments in August 2019

The results of the SQL query executed in Figure 1 was stored to disk as a csv file. The downloaded csv file contained 5847 entries and the following two fields:

1. Comment\_date <string>
2. Comment <string>

The *Comment*\_*date* field contains the date on which the comment was made, and the *Comment* field contains the actual comment.

## Proposed high-level solution design

The proposed technical solution to answering this question is implemented by ways of Apache Spark. Spark has the capability to expose a rich and complex ingestion framework whereby data sources from various shapes, sizes and types can be ingested. The first solution that the author describes is the actual solution implemented and that has the capability to ingest data from csv files, apply data cleansing and data wrangling techniques, apply NLP processing and apply visualization to the processed data.

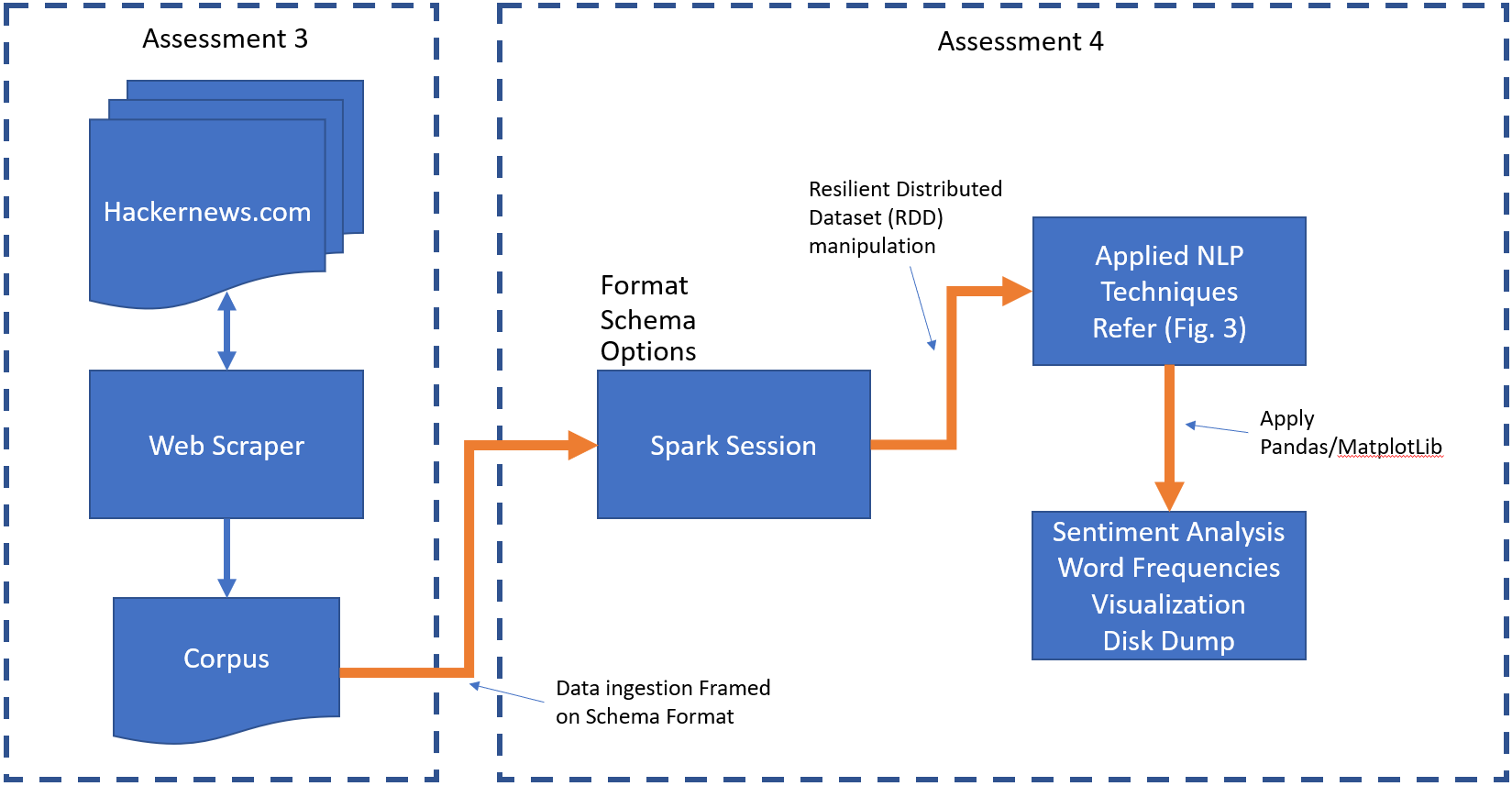


Figure 2 - Basic Implementation of a Spark solution

Figure 2 above, illustrates the basic components of the Spark solution that deliver the results as reported later in this document. The left dashed box labelled, Assessment 3, represents the components that implemented the scraping solution in Assessment 3. The output of that exercise was the document object labelled, Corpus, at the bottom of the left-side dashed-box. The scraped corpus consists of 930 records and each record has the following fields:

1. Index
2. ID
3. Link\_title
4. Web\_Link
5. Points

The data in the corpus is sorted in an ascending order according to the points field. The corpus, containing all 4 of the above fields are then ingested into a Spark session, which was custom created for the ingestion. To realise this ingestion, the corpus is overlayed with a pre-defined Schema. Once the corpus is ingested and framed by the specified schema it is maintained and managed as a Resilient Distributed Dataset (RDD). The schema that is used to frame the content of the corpus matches the number of fields and their associated types as specified for the corpus.

Once the corpus has been ingested, member functions of the Spark RDD object are called to assist in doing some basic house cleaning on the data. Once the data is suitably prepped, some additional helper functions are called via the RDD’s map function to perform several NLTK preparations on the remainder of the corpus. This process is illustrated in Figure 3. From Figure 3 it can be observed that the following helper functions are applied to the corpus:

1. LOWER
2. FILTER
3. SENTENCE TOKENIZE
4. WORDS TOKENIZE
5. PUNCTUANTION
6. LAMMETIZATION
7. RE-JOIN

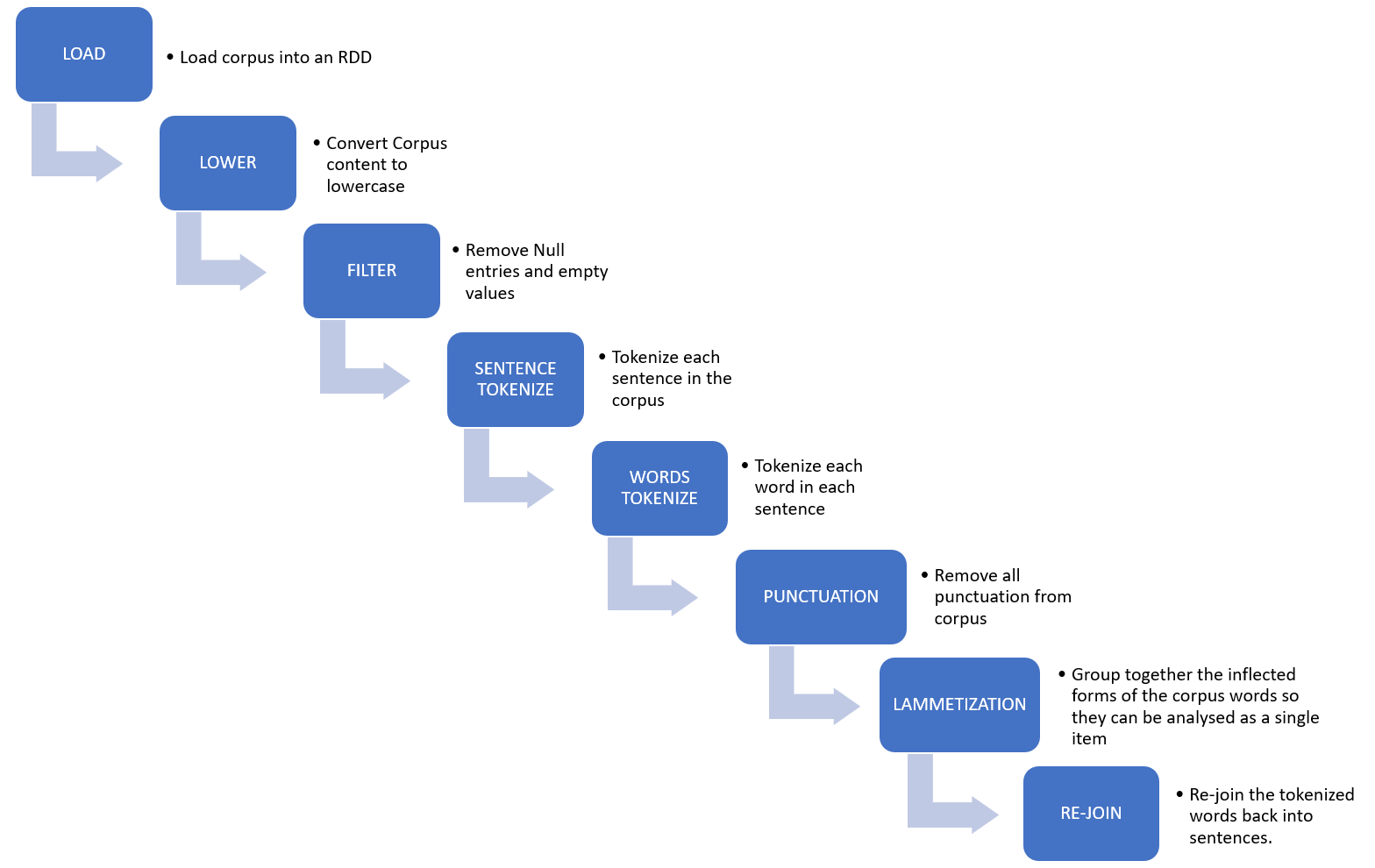


Figure 3 - NLP process applied to the Hacker News Corpus

Each of the above function’s description is listed in Figure 3 alongside the box’s depicting the applied functions. The result of the process illustrated in Figure 3 is a word corpus that has been preparade for further analysis and NLP application.

*Note: At this point the author needs to point out that he leaned heavily on the NLP examples as was provided in Week 5’s practical tutorials to shape the preparation steps as depicted in* Figure 3*. There was really no need to re-invent the wheel as far as corpus preparation goes.*

Figure 2 only illustrates and describes the most basic steps required to ingest data on a once-off basis. This basic ingestion patterns can, however, be extended into something much richer and functionally advanced. Therefore, consider the process depicted in Figure 4, which extends the ingestion patterns in Figure 3 in several ways. A word on the Apache Spark technologies before we dive into the intricacies of the advanced ingestion pattern depicted in Figure 4:

*“Apache Spark is a lightning-fast cluster computing designed for fast computation. It was built on top of Hadoop MapReduce and it extends the MapReduce model to efficiently use more types of computations which includes Interactive Queries and Stream Processing.*(“Spark Streaming—Spark 2.4.4 Documentation,” n.d.)*”*

I could not have stated it better myself. The emphasis, however, falls on the words – “Stream Processing” in the above blurb from the official online Spark documentation. Extending the basic ingestion pattern firstly focusses on an additional interface that is needed between the source of the data and the Spark Session instance. In Figure 4 this interface is depicted by a dashed box labelled, Data Collection. The data collection interface can be implemented with various contemporary product offerings currently available. Three examples of these offerings are illustrated in Figure 4 namely, Kafka, Flume and Kinesis.

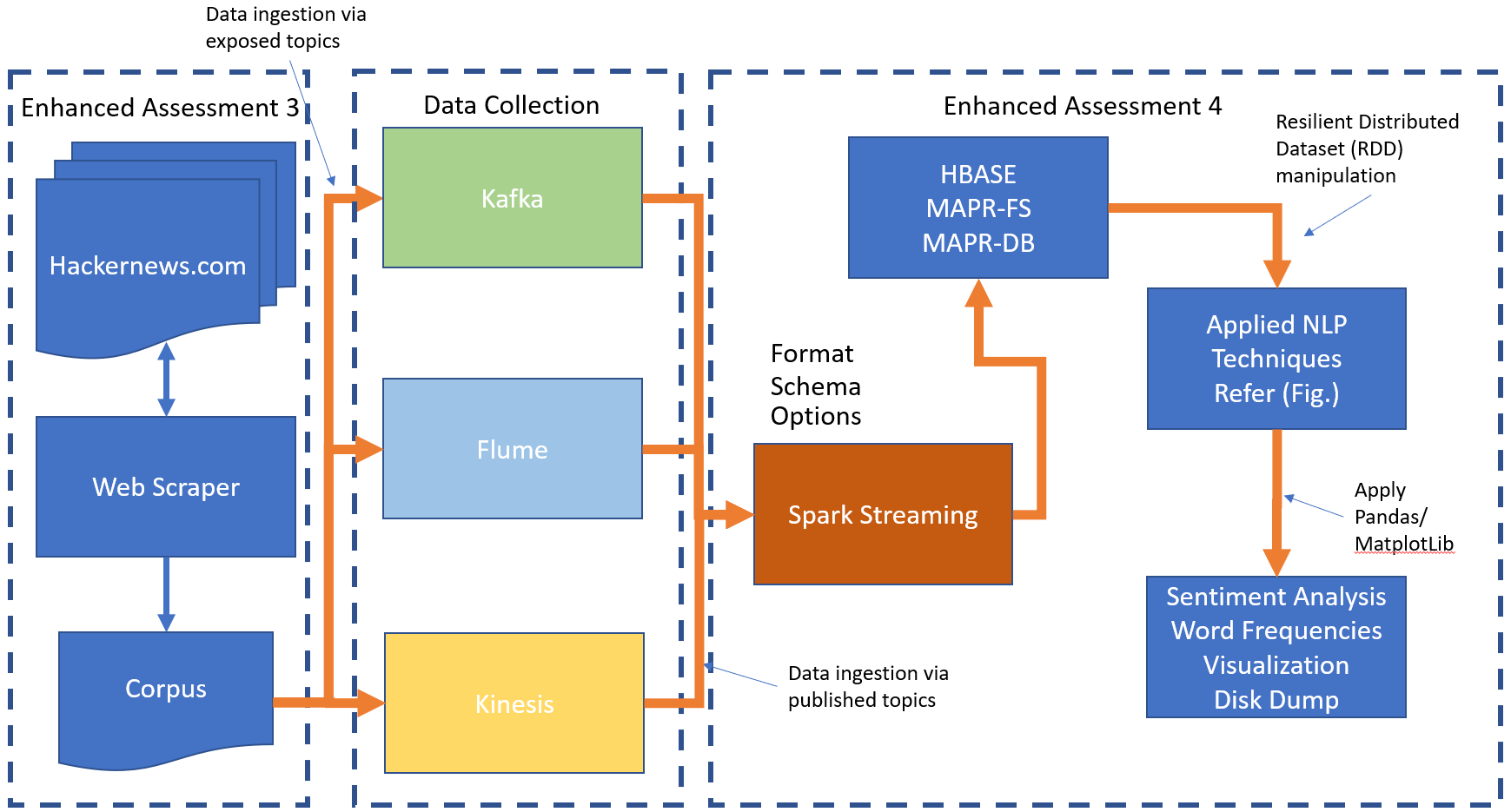


Figure 4 - Advanced Implementation of a Spark solution

Kafka, an Apache offering, can support data streams for multiple applications, whereas Flume, also an Apache offering, is specific for Hadoop and big data analysis. Kafka can process and monitor data in distributed systems whereas Flume gathers data from distributed systems to land data on a centralized data store. Kafka is mainly a distributed *publish-subscribe* messaging system that receives data from disparate source systems and makes the data available to target systems in real time. Kinesis is an Amazon Web Service (AWS) for processing big data in real time. Kinesis is capable of processing hundreds of terabytes per hour from high volumes of streaming data from sources such as operating logs, financial transactions and social media feeds. The *publish-subscribe* capability of Kafka is the main mechanism through which sources connect to a pre-defined interface that negotiates the flow of data and exposes the data through a pre-defined topic to the Spark Stream instance.

Thus, the Data Collection interface in Figure 4 deals effectively with the streaming portion of data. This is an invaluable capability as real-time analysis has become highly important in order for businesses to make strategic decisions in real-time on insights gained from analytical systems that are fed live data by ways of above described streaming technologies.

The dashed box labelled, Enhanced Assessment 4, depicts the advanced offering for the data processing side of the streaming line. Here you’ll notice that we replaced the *Spark* instance with the *Spark Streaming* instance. The Key difference between the *Spark* and *Spark Stream* instances is that *Spark* uses RDD abstraction, *Spark**Stream* instead uses the concept of DStream which is basically RDD separated with a batch interval. As a result, you get an output with computations you did at the end of every interval as opposed to at the end of processing an entire single data load.

By implementing the advanced ingestion process depicted in Figure 4, it will become possible to harvest headlines as posted by users on the Hacker News forum. Obviously, the scraper functionality has to be altered to cyclically poll the Hacker News forum for new posts and also have the ability to subscribe to published Kafka topics that are aligned with the correct Spark Stream instance on the other side. The Spark Stream instance has the capability to utiliz numerous underlying data storage technologies, which can range from HBASE, to MAPR-FS and MAPR-DB. Once the data resides in any of the latter named storage technologies, Spark is utilized to apply any number of NLP techniques such as listed in Figure 3, which facilitates the extraction of insights such as sentiment analysis, word frequency, etc.

For the basic implementation of this assessment, the author has opted to create visualizations with Tableau. Tableau also have the capability to visualize dynamic data, especially when the velocity is high. Therefore, real-time analytics can be realised via this capability of Tableau.

## Analysis

Your analysis tables and graphs. This should be interspersed with commentary so that it can be read as a document. Add appropriate interpretations and discussions of your results and model selection/performance where appropriate. The codebase and necessary resources should be shared to the reader via a repository link with at least five (5) commits as evidence of code management.

## Discussion

Your conclusions from your project. Restate the original objectives and/or problems and contrast this against the obtained achievements. Discuss the limitations of the analysis, such as what you were able to show or what you couldn't show. Include suggestions for further work, such as including other data sources that might be useful for a future analysis and/or more things you could have done if given more time.

# In this task, you will also be assessed for your writing, in that the report must be:

## References

Hacker News. (2019). In *Wikipedia*. Retrieved from https://en.wikipedia.org/w/index.php?title=Hacker\_News&oldid=924206938

Konnikova, M. (2014, December 17). *How Headlines Change the Way We Think*. Retrieved from https://www.newyorker.com/science/maria-konnikova/headlines-change-way-think

Spark Streaming—Spark 2.4.4 Documentation. (n.d.). Retrieved November 28, 2019, from https://spark.apache.org/docs/latest/streaming-programming-guide.html

Surber, J. R., & Schroeder, M. (2007). Effect of prior domain knowledge and headings on processing of informative text. *Contemporary Educational Psychology*, *32*(3), 485–498. https://doi.org/10.1016/j.cedpsych.2006.08.002

Tan, L. I., Phang, W. S., Chin, K. O., Alfred, R., & Anthony, P. (2014). *Impact of financial news headline and content to market sentiment*. https://doi.org/10.7763/IJMLC.2014.V4.418