# Cloud-Based Sentiment Analysis for Measuring Customer Satisfaction in the Moroccan Banking Sector Using Naïve Bayes and Stanford NLP

Submitted: 26th June 2019; accepted: 25th March 2020

*Anouar Riadsolh, Imane Lasri, Mourad ElBelkacemi*

DOI: 10.14313/JAMRIS/4-2020/47

**Abstract:** *In a world where every day we produce 2.5 quintillion bytes of data, sentiment analysis has been a key for making sense of that data. However, to process huge text data in real-time requires building a data processing pipeline in order to minimize the latency to process data streams. In this paper, we explain and evaluate our proposed real-time customer' sentiment analysis pipeline on the Moroccan banking sector through data from the web and social network using open-source big data tools such as data ingestion using Apache Kafka, In-memory data processing using Apache Spark, Apache HBase for storing tweets and the satisfaction indicator, and ElasticSearch and Kibana for visualization then NodeJS for building a web application. The performance evaluation of Naïve Bayesian model show that for French Tweets the accuracy has reached 76.19% while for English Tweets the result was unsatisfactory and the resulting accuracy is 56%. To remedy this problem, we used the Stanford core NLP which, for English Tweets, reaches a precision of 80.7%.*

**Keywords:** *Big Data Processing; Apache Spark; Apache Kafka; Real-time Text Processing; Sentiment Analysis; Stanford core NLP; Naïve Bayes classifier*

## 1. Introduction

Sentiment analysis, or opinion mining, has obtained much attention in recent years with the advent of various social networking like Twitter which is a popular microblogging service where users gives their opinions. The aim of sentiment analysis is to define automatic tools able to extract subjective information from texts in natural language, such as opinions and sentiments, to create a structured and actionable knowledge to be used by either a decision support system or a decision maker [1, 3].

With the rise of Big Data and data lake, machine learning is taking on a whole new dimension as businesses now have access to a huge amount of different variables that, when correlated, can be an extremely powerful decision asset. That why the Moroccan banking sectors are particularly interested in using various machine learning algorithms for the sentiment analysis and it wants to go beyond the statistical approach to predict changes in financial markets through the analysis of Tweets, predicting that a customer will leave his bank, detecting fraud, improving customer' satisfaction. To achieve these goals, we propose two methods to analyze the great numbers of data available on Twitter and distribute them into three categories (positive, negative or neutral). This is very useful because it allows banks to improve services of products, marketing, customer service, business performance, risk management and to calculate indicators (KPIs). The first method is based on a Stanford CoreNLP API [2] for English Tweets and the second method is based on Naïve Bayes classifier [4] for French Tweet.

In this paper, we have proposed and evaluated a real-time processing pipeline using the open-source tools in Microsoft Azure that can capture a large amount of data efficiently. Our suggested system uses Apache Kafka [5] as data ingestion system, Apache Spark [6] as a real-time data processing system, Apache Hbase [7] for persistent distributed storage, and ElasticSearch and Kibana for visualization.

Traditionally, Apache Kafka accepts incoming data and sends it to Apache Kafka broker rapidly. Then Apache Spark consumes the data and performs predictive analytics using Spark's MLib module and Stanford Core NLP. Finally, we use Apache Hbase to store the data. We developed a simple visualization component to analyze the results using Kibana and ElasticSearch.

The rest of the paper has six more sections and ordered as follows. In Section 2 we introduce related work on sentiment classification. And in section 3 we describe the data preprocessing steps. In Section 4 we explain the system components to build the real-time sentiment analysis pipeline. Section 5 presents the two methods to analyze sentiment. Finally, in Section 6 and 7 we discuss the results and future extensions of our work.

## 2. Related Work

In recent years, the growth of social media networks has made sentiment analysis a popular area of research. Accordingly, several recent articles have focused on the analysis of the media sentiment for a variety of purposes such as the public opinion and prediction of election results.

Shoro et al. [8] extract and process some meaningful information from some sample big data source, such as Twitter, using Apache Spark streaming. Bollen

et al. [9] analyzed Tweets to find correlation between overall public mood and social, economic and other major events. The authors extracted six mood states (anger, tension, depression, vigor, confusion, fatigue) from the Tweets then they compared the results to a record of popular events gathered from media and sources. In [10], the authors used two supervised machine learning algorithms: K-Nearest Neighbour(K-NN) and Naïve Bayes to facilitate the quick discovery of sentimental contents of movie reviews and hotel reviews on the web.

Our work differs in the scale of the Twitter dataset size as well as the number of banks we analyze. Our objectif, in this paper, is to build a real-time sentiment analysis pipeline on Microsoft Azure using open-source big data tools. In addition, we look at customer' sentiment towards banks. Finally, we derive some meaningful insights from this analysis and demonstrate the value of such analysis in order to measure the customer' satisfaction and to assess the impact of banks on social media.

## 3. Real-Time Sentiment Analysis Pipeline

The customer' sentiment analysis on the Moroccan banking sector requires a collection of data representing the Moroccan citizens' opinion of the services offered by the banks. For this, we extracted the data from Twitter, in real-time through Apache Kafka then processed by Spark Streaming, the result is then stored on Apache Hbase in order to visualize the indicators and measure customer' satisfaction on Kibana through the Elasticsearch search engine. Figure 1 shows the proposed real-time processing pipeline architecture :

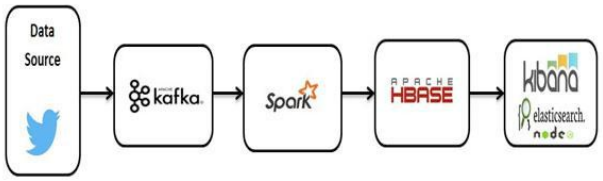


**Fig. 1.** Proposed data analytics pipeline architecture

In the rest of this section, we briefly explain each of these components in turn.

### 3.1. Twitter Data Extraction

With the rise of the internet and mobile telecommunications the need and importance of extracting data from the web is becoming increasingly loud and clear. In fact, online social networks attract the most users, though users of these new technologies provide their data through multiple sources. Of those networks, Twitter makes these data relatively easy to obtain by providing users' data through two APIs, the streaming API and the REST API, that we used in this project. And it allows us to retrieve data from each Tweet as a JavaScript Object Notation (JSON) document via OAuth which is required for API authentication.

Nonetheless, because of Twitter's popularity, there are a large number of software libraries to access Twitter, including in Python, R and Spark.

### 3.2. Data Ingestion Using Apache Kafka

Nowadays businesses collect large volumes of structured and unstructured data, in order to discover real-time insights that inform decision making and support digital transformation [3]. Data ingestion is the process of importing, loading, processing data and storing them in a database. It requires fetching data from a variety of sources, such as social media sites, web logs, RDBMS and streaming data.

There are many tools available today for data ingestion such as Apache Kafka, Apache Flume [11] and RabbitMQ [12].

We used, in this paper, Apache Kafka as a data ingestion system. It is a distributed publish-subscribe messaging system where multiple producers publish the message on a topic and multiple consumers consume the messages by subscribing to that topic. Kafka uses ZooKeeper [13] internally or externally to do leadership election of Kafka broker, topic partition pairs, managing service discovery for Kafka brokers. The terms mentioned below are used in Apache Kafka:

- **Topic**: it identifies a class of messages.
- **Partition**: it is used to scale a topic across many servers.
- **Producer**: pushes messages to topics.
- **Consumer**: pulls messages from topics.
- **Broker**: an instance of the Kafka service.

### 3.3. Parallel Data Processing With Apache Spark

There are multiple parallel and distributed tools such as Apache Hadoop [14], Apache Spark, and Apache Storm [15]. We used Apache Spark as a real-time in-memory processing system. It was developed by the University of Berkeley to solve the limitations of Apache Hadoop. It include libraries for SQL, streaming, machine learning and graph processing engines. The resilient distributed datasets (RDD's) [16] is the abstract data type for distributed and parallel computing for Apache spark.

Specially, the SparkContext allocate resources across applications. Spark use executors to run computations and store data for the application. Then, it sends the application code to the executors. Finally, SparkContext sends tasks to the executors to run. Figure 2 below illustrates the execution architecture of Apache Spark application.

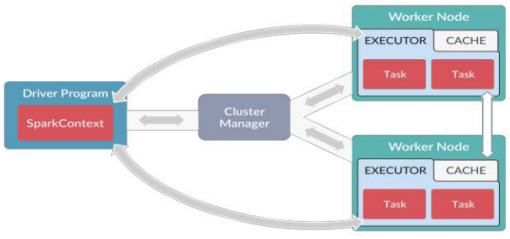


**Fig. 2.** Execution Flow of Apache Spark application

### 3.4. Distributed Data Storage on Apache HBase

The NoSQL databases [17] system are non-relational, distributed database system that enable the fast analysis of high-velocity data with disparate data types. The key factors of the NoSQL databases system is the scalability, high availability, and fault tolerance. There are many of NoSQL databases like Apache Hbase, Mongodb [18] and Apache Cassandra [19]. We use in our project Apache Hbase as NoSQL distributed database for the real-time system.

HBase is a Java-based, open source, NoSQL, non-relational, column-oriented, distributed database built on top of the Hadoop Distributed Filesystem (HDFS) [20], modeled after Google's BigTable paper. HBase brings to the Hadoop ecosystem most of the BigTable capabilities. HBase is built to be a fault-tolerant application hosting a few large tables of sparse data (billions/trillions of rows by millions of columns), while allowing for very low latency and near real-time random reads and random writes [21]. This is the major terms used in Apache Hbase:

- **Table:** In HBase the data is organized into tables. Table names are strings.
- **Row**: In each table the data is organized in rows. a line is identified by a unique key (RowKey). The Rowkeys does not have a type, it is treated as an array of bytes.
- **Column** Family: Data within a row is grouped by column family. Each row of the table has the same column family, which are defined when the table is created in HBase. The names of the column family are strings.
- **Column qualifier**: It allows access to data within a column family. Like rowkeys, the column qualifier is not typed, it is treated as an array of bytes.
- **Cell**: The combination of RowKey, Column Family and Column Qualifier uniquely identifies a cell. The data stored in a cell is called the values of that cell. Values have no type, they are always considered as byte array.
- **Version**: Values within a cell are versioned. The versions are identified by their timestamp. The number of versions is configured through the Column Family. By default, this number is equal to three.

HBase is composed of 3 types of servers in a master slave type of architecture. Region servers serve data for reads and writes. Region assignment and DDL operations are handled by the HBase Master. Zookeeper, which is part of HDFS, maintains a live cluster state. The NameNode maintains metadata information for all the physical data blocks that comprise the files. Figure 3 shows the architecture of Apache Hbase:

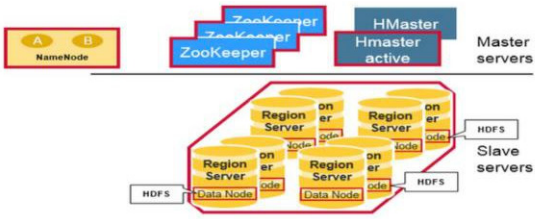


**Fig. 3.** Hbase Architecture

### 3.5. Visualization Component

The visualization displays the real-time dashboard on Kibana based on a real-time processed data stored on Elasticsearch, which helps in both decision making and visualization purposes. For, Twitter Sentiment Analysis Application, we created a web application in node.js on Microsoft Azure that display graphs and geo map of positive, negative and neutral Tweets. The web application also allows users to search a particular bank and finds its occurrence in positive, neutral and negative Tweets. The web application also show the customer' satisfaction rate.

## 4. Data Preprocessing

Sentiments analysis requires many preprocessing steps such as tokenization, stop words removing, and stemming. These steps approximately consumes 80% of the time and efforts. The preprocessing steps play, in addition to their preparation role, the data reduction role by excluding worthless feature from the bag of words.

- **Lower uppercase letters:** Is the first step in the preprocessing wich consist on changing uppercase letter to lowercase letter, so that the analysis will not be case sensitive.
- **Remove numbers:** Numbers can be removed from the sentences because they are not significant in the data analysis. Accordingly, this preprocessing step use regular expression.
- **Remove URLs and user references:** a twitter user can use a nominated user @username to mention another user in Tweets, send him a message, or link to his profile. Therefore it is necessary to remove symbols such as '@' and '#' from the words because they are not relevant for analyzing the content of a text.
- **Remove stop words:** Stop words are words which are commonly used in a language and is not useful for the analyzing system. The stop words of English and french language are stored in a list in HDFS. In this step, items in the tokens liste compared with each word in the stop word table in order to delete the stop words from tokens table for each Tweet.

- **Tokenize:** Tokenization annotator splits sentences into smaller units called token which can be words, numbers, n-grams, and symbols. So the token is created by splitting the sentence on each space.
- **Detect POS tags:** Part of speech tagging task aims to assign every word/token in plain text a category such as noun, verb, adjective, etc.
- **Lemmatize:** When processing samples, "word" and "words" would be considered as two different features. Hence, in order to improve the features reduction process, the unigrams can be lemmatized. This preprocessing step mainly allows to remove plurals and conjugations.

## 5. Material and Methods

The main goal of the research is to analyze customer' sentiment about Moroccan banks from Twitter. We used two methods to classify Tweets into 3 categories ('Positive', 'Negative' or 'Neutral'). The first approch is based on a Stanford CoreNLP API for English Tweets and the second approch is based on Naïve Bayes classifier for French Tweet because Stanford NLP API does not contain sentiment models for languages excepting English. In this section, we explain this two methods.

### 5.1. Methodology

*1) Stanford CoreNLP*

Natural language processing (NLP) is a branch of artificial intelligence focused on the interactions between human language and computers. Nowadays, many tools have been published to do natural language processing jobs. Stanford CoreNLP is a great Natural Language Processing (NLP) tool for analysing text. Given a paragraph, CoreNLP splits it into sentences then analyzes it to return the base forms of words in the sentences, their dependencies, parts of speech, named entities and many more.

Stanford CoreNLP only supports English for sentiment analysis. It is implemented in Java and our main code-base is written in Spark using Scala that why we access it through an API. This model proposes many linguistic analysis tools like:

- **The part-of-speech (POS) tagger:** is a process of associating the words of a text with a corresponding type.
- **The Named-Entity Recognition (NER):** allows to recognize in a text a certain type of categorizable concepts such as names of people, names of organizations or companies, names of places, quantities, distances, values, dates, etc.
- **Parser:** It consists in highlighting the structure of a text. This structure is often a hierarchy of syntagms, represented by a syntax tree or tagged tree ParseTree) whose nodes can be equipped with additional information.
- **Sentiment Analysis:** Predicts the sentiment of a text (positive, negative, neutral) but based on a new type of recursive neuron network that relies on grammatical structures.

Stanford sentiment analysis calculates the sentiment by relying on how words make up the meaning of the text. This applies by introducing the Stanford Sentiment Treebank which is a dataset with fully labeled analysis trees that allow complete analysis. The dataset is based on data introduced by Pang and Lee (2005) [22] and includes 11,855 sentences taken from film reviews. These sentences were analyzed with the Stanford parser which itself includes a total of 215 154 sentences, each annotated by 3 human judges. This new dataset allows us to analyze sentiment with subtlety.

Algorithm 1

*Input*: a text *t*

*Output*: a predicted class

*sf* ∈ {'Positive','Negative','Neutral'}

**Steps:**

*a)* Segmentation of the text into a sentence with the annotator ssplit by detecting the points.

*b)* Fragmentation of sentences using the Tokenize annotator into smaller units called token which can be words, n-grams (group of n consecutive tokens), numbers, symbols, and punctuation.

c) Association of a morphosyntactic class for each fragment according to its context

*d)* Application of the lemma annotator to the words of the text in order to detect to which family belongs the word and replace it with its canonical form which is the infinitive if it is a verb and masculine singular for other words.

*e)* Application of the parsetree annotator for a syntactic analysis distributed in the form of a tree with two main branches corresponding to the nominal sentence and to the verbal sentence if it exists.

*f)* Using Sentiment Annotator to Attache a binarized tree of the sentence to the sentence level CoreMap. The nodes of the tree then contain the annotations from RNNCoreAnnotations indicating the predicted class and scores for that subtree.

*g)* Calculating the weight of the sentiment ws in each sentence which is equal to the sentiment s multiplied by the size of the sentiment sz:

$$ws = s \,.\, sz \tag{1}$$

*h)* In order to calculate of the weight w of the sentiment of the whole text we divide the sum of the weight of the sentiment ws of the whole sentences in the text by the size of the text st:

$$w = \Sigma ws \,/\, st \tag{1}$$

*i)* If $0 < w < 2$ then *sf* = 'Negative'

Else if $3 < w < 5$ then *sf* = 'Positive'

Else *sf* = 'Neutral'

*2) Naïve Bayes Classifier*

Naïve Bayes Classifier is a popular algorithm in Machine Learning. It is a supervised classification algorithm that depend on Bayes' theorem which is based on conditional probabilities. The basic idea is to find the probabilities of categories given a text by using the joint probabilities of words and categories. It is based on the assumption of word independence. The starting point is the Bayes' theorem for conditional probability, stating that, for a given data point *x* and class C:

$$P(Cj / xi) = (P(xi/Cj) . P(Cj)) / P(xi) \quad (1)$$

Cj represent jth class of classes{1,2,3..n}
xi represent features vector of ith sample of samples{1,2,3..m}

We used Naïve Bayes with Apache Spark MLlib for Text classification. It takes an RDD of LabeledPoint and an optional smoothing parameter lambda as input, an optional model type parameter (default is "multinomial"), and outputs a NaiveBayesModel, which can be used for evaluation and prediction.

Algorithm 2

*Input*: a text *t*, a list of stop words *sw*
*Output*: a predicted class
*s* ∈{'Positive','Negative','Neutral'}

**Steps:**

a) Removing stop words, links, emails and worthless feature from the text.

b) Transforming the text into linear vectors

c) Training the model on a dataset that shows the expected class which is the polarity of the text in relation to the features.

d) Classification of the predicted polarity *p*:
If *p* = 0 then *s* = 'Negative'
Else if *p* =4 then *s* = 'Positive'
Else *s* = 'Neutral'

### 5.2. Customer' Satisfaction Rate Calculation

Key performance indicators (KPIs) wich are like milestones on the road to online retail success. Ecommerce entrepreneurs identify progress toward sales, marketing, and customer' service goals monitoring them.

We have calculated in this research the customer' satisfaction rate by bank which is equal to the sum of the positive Tweets for a bank multiplied by 100 and divided by the total number of Tweets. Figure 4 shows the customer' satisfaction rate by bank stored in Hbase:

| banque | tauxsatisfaction | date |
|---|---|---|
| BMCI Bank | 15.686274509803921 | 2018-05-28 04:33:02 |
| Al Akhdar Bank | 20.0 | 2018-05-28 04:33:02 |
| CFG Bank | 9.090909090909092 | 2018-05-28 04:33:02 |
| Crédit du Maroc | 50.0 | 2018-05-28 04:33:02 |
| Bank Al-Maghrib | 0.0 | 2018-05-28 04:33:02 |
| Arab Bank Maroc | 75.0 | 2018-05-28 04:33:02 |
| BTI Bank | 75.0 | 2018-05-28 04:33:02 |
| Bank Assafa | 0.0 | 2018-05-28 04:33:02 |
| CIH Bank | 50.0 | 2018-05-28 04:33:02 |
| BMCI Najmah | 33.33333333333333 | 2018-05-28 04:33:03 |
| Banque populaire ... | 11.235955056179774 | 2018-05-28 04:33:03 |
| Bank Al Yousr | 0.0 | 2018-05-28 04:33:03 |
| Attijariwafa Bank | 0.0 | 2018-05-28 04:33:03 |
| Société générale ... | 0.0 | 2018-05-28 04:33:03 |
| Arreda | 75.0 | 2018-05-28 04:33:05 |
| Umnia Bank | 57.14285714285714 | 2018-05-28 04:33:05 |
| Crédit Agricole d... | 100.0 | 2018-05-28 04:33:05 |
| BMCE Bank | 38.095238095238095 | 2018-05-28 04:33:05 |
| Dar Al-Amane | 20.0 | 2018-05-28 04:33:05 |

**Fig. 4.** The customer' satisfaction rate by bank

## 6. Experimental Results

### 6.1. Performance Evaluation

Accuracy is the ratio of number of correct predictions to the total number of input samples.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions\ made}$$

Our performance evaluation show that for French Tweets the accuracy of Naïve Bayesian model has reached 76.19% while for English Tweets the resulting accuracy is 56%. That why we used the Stanford core NLP which, for English Tweets, reaches a precision of 80.7%.

### 6.2. Result

In the rest of this section, we present the visualizations of the result on Kibana. We first visualize the data in the form of heat map which express the number of Tweets according to the sentiment deducted for each bank. Figure 5 shows the occurrence of banks in positive, neutral and negative Tweets.


**Fig. 5.** The number of Tweets by bank according to the sentiment

The second graph in figure 6 is a histogram expressing the customer' satisfaction rate with respect to each bank.


**Fig. 6.** Customer' satisfaction by bank

The figure 7 is a pie chart that shows the rate of positivity, neutrality and negativity of customer' opinion of the Moroccan banking sector.

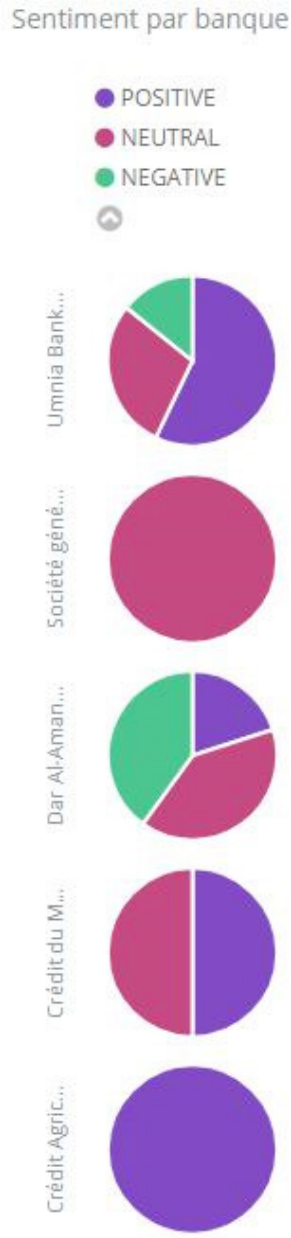


**Fig. 7.** A pie chart for the rate of positivity, neutrality and negativity by bank

The fourth graph in the figure 8 is a map explaining the number of Tweets by region. Having a large number of Tweets, this explains that this region is more banked than any other. This graph gives an idea of the regions that banks must target to improve financial inclusion.


**Fig. 8.** A map for the number of Tweets by region

The figure number 9 and 10 below represent the number of Tweets by banks and the numbers of Tweets by region.

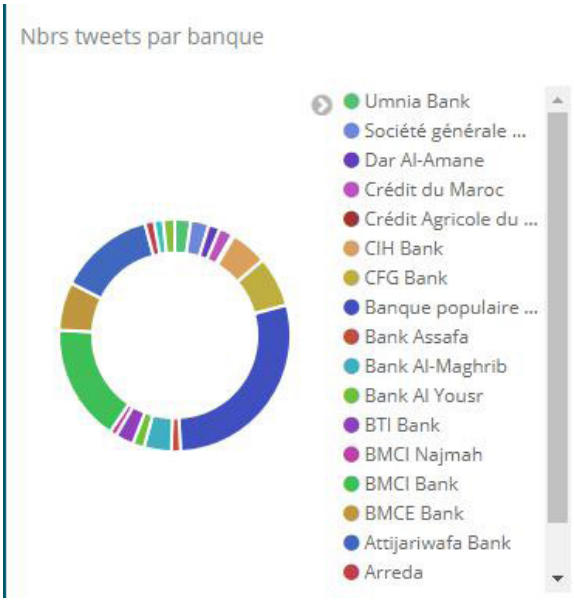


**Fig. 9.** Number of Tweets by bank

Nbr des tweets par ville

| location: Descending | Nbr des tweets |
|---|---|
| Tétouan, Royaume du Maroc | 3 |
| Toulouse, Midi-Pyrénées | 1 |
| Tiznit, Royaume du Maroc | 1 |
| Tigery | 1 |
| Taza, Royaume du Maroc | 2 |
| Tanger Médina | 1 |
| Shanghai, People's Republic of China | 1 |
| Settat, Royaume du Maroc | 1 |

**Fig. 10.** Number of Tweets by region

The Kibana Dashboard page is where we combined multiple visualizations onto a single page, then filter them by providing a search query or by selecting filters by clicking elements in the visualization. Figure 11 shows the Kibana dashboard which contain all the visualizations mentioned above.

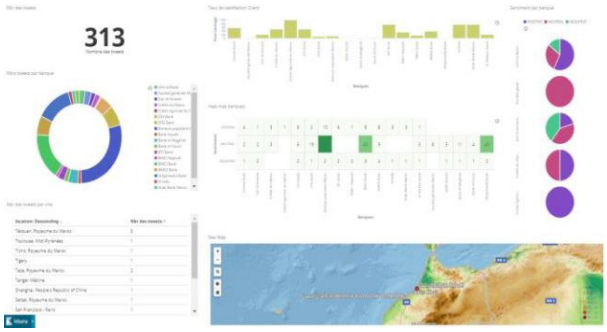


**Fig. 11.** The sentiment Analysis Dashboard

## 7. Conclusion

In this paper, we analyzed the sentiment of customer on Twitter towards 23 banks in Morocco. To the best of our knowledge, no other work has attempted to analyze sentiment of users towards Moroccan banks. We first began by studying the business needs for the types of data, the results it wants and the right tools. Then we analyzed the customer' sentiment using two methods to classify Tweets into 3 categories ('Positive', 'Negative' or 'Neutral'). The first method is based on a Stanford CoreNLP API for English Tweets and the second method is based on Naïve Bayes classifier for French Tweet. However performing such Twitter sentiment analysis against the large amount and high velocity of data requires large-scale processing of data on multiple machines using big data tools. The real-time system we proposed and evaluated in this paper is able to perform the Twitter sentiment analysis over large and high velocity of data near the real-time.

In future, we will work on setting up an analysis solution that predicts yield and turnover, while tracking multiple features. We would also like to compare try and come up with an efficient sentiment analyzer like random forest, Support vector Machine etc.

## Acknowledgements

The authors would like to thank Bank of Maghreb and Faculty of Sciences for the support during this research work.

## AUTHORS

**Anouar Riadsolh*** – Laboratory of Conception and Systems, Faculty of Sciences, Mohammed V University in Rabat, Morocco, e-mail: anouarriadsolh@yahoo.fr.

**Imane Lasri** – Laboratory of Conception and Systems, Faculty of Sciences, Mohammed V University in Rabat, Morocco, e-mail: imanelasri95@gmail.com.

**Mourad ElBelkacemi** – Laboratory of Conception and Systems, Faculty of Sciences, Mohammed V University in Rabat, Morocco, e-mail: mourad_prof@yahoo.fr.

*Corresponding author

## REFERENCES

[1] F. A. Pozzi, E. Fersini, E. Messina and B. Liu, *(eds.) Sentiment analysis in social networks*, Morgan Kaufmann, 2017, DOI: 10.1016/C2015-0-01864-0.

[2] C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard and D. McClosky, "The Stanford CoreNLP Natural Language Processing Toolkit". In: *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014, 55–60, DOI: 10.3115/v1/P14-5010.

[3] A. Riadsolh and M. E. Belkacemi, "Toward a Good Decision to Improve the Weight of Control of Expenditure for Local Communities", *American Journal of Applied Sciences*, vol. 13, no. 3, 2016, 299–306, DOI: 10.3844/ajassp.2016.299.306.

[4] K. Ming Leung, "Naive Bayesian Classifier", Polytechnic University, 2007, https://cse.engineering.nyu.edu/~mleung/FRE7851/f07/naiveBayesianClassifier.pdf. Accessed on: 2021-02-05.

[5] "Apache Kafka: a distributed straming platform", kafka.apache.org. Accessed on: 2021-02-05.

[6] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker and I. Stoica, "Spark: Cluster Computing with Working Sets". In: *Proceedings of the 2nd USENIX Conference on Hot Topics in Cloud Computing (HotCloud'10)*, 2010.

[7] "Apache HBase ™ Home", https://hbase.apache.org. Accessed on: 2021-02-05.

[8] A. G. Shoro and T. R. Soomro, "Big Data Analysis: Apache Spark Perspective", *Global Journal of Computer Science and Technology*, 2015.

[9] J. Bollen, H. Mao and A. Pepe, "Modeling public mood and emotion: Twitter sentiment and socio-economic phenomena". In: *ICWSM*, vol. 11, 2011, 450–453.

[10] L. Dey, S. Chakraborty, A. Biswas, B. Bose and S. Tiwari, "Sentiment Analysis of Review Datasets Using Naïve Bayes' and K-NN Classifier", *International Journal of Information Engineering and Electronic Business (IJIEEB)*, vol. 8, no. 4, 2016, DOI: 10.5815/ijieeb.2016.04.07.

[11] "Welcome to Apache Flume — Apache Flume", http://flume.apache.org/. Accessed on: 2021-02-05.

[12] A. Videla and J. J. W. Williams, *RabbitMQ in Action*, Manning, 2012.

[13] "Apache ZooKeeper", https://zookeeper.apache.org. Accessed on: 2021-02-05.

[14] O. O'Malley, "Terabyte sort on apache hadoop", 2008, sortbenchmark.org/YahooHadoop.pdf. Accessed on: 2021-02-05.

[15] A. Toshniwal, S. Taneja, A. Shukla, K. Ramasamy, J. M. Patel, S. Kulkarni, J. Jackson, K. Gade, M. Fu, J. Donham, N. Bhagat, S. Mittal and D. Ryaboy, "Storm@twitter". In: *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data*, 2014, 147–156, DOI: 10.1145/2588555.2595641.

[16] M. Zaharia, M. Chowdhury, T. Das, A. Dave, J. Ma, M. McCauley, M. J. Franklin, S. Shenker and I. Stoica, "Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing". In: *Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation*, 2012.

[17] H. Jing, E. Haihong, L. Guan, and D. Jian, "Survey on NoSQL database". In: *2011 6th International Conference on Pervasive Computing and Applications*, 2011, 363–366, DOI: 10.1109/ICPCA.2011.6106531.

[18] K. Chodorow, "Introduction to MongoDB", https://archive.fosdem.org/2010/schedule/events/nosql_mongodb_intro.html. Accessed on: 2021-02-05.

[19] "Apache Cassandra", https://cassandra.apache.org/. Accessed on: 2021-02-05.

[20] K. Shvachko, H. Kuang, S. Radia and R. Chansler, "The Hadoop Distributed File System". In: *Proceedings of the 2010 IEEE 26th Symposium on Mass Storage Systems and Technologies (MSST)*, 2010, 1–10, DOI: 10.1109/MSST.2010.5496972.

[21] J.-M. Spaggiari and K. O'Dell, *Architecting HBase Applications: a Guidebook for Successful Development and Design*, O'Reilly Media, 2016.

[22] B. Pang and L. Lee, "Seeing stars: exploiting class relationships for sentiment categorization with respect to rating scales". In: *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, 2005, 115–124, DOI: 10.3115/1219840.1219855.