uc3m | Universidad Carlos III de Madrid

Grado Universitario de Ingeniería Informática
2018-2019

Trabajo Fin de Grado

"MRT: Plataforma de análisis y visualización de sentimientos en Twitter"

Alejandro Corrochano Navarro

Tutor

José Antonio Iglesias Martínez

Leganés, Julio de 2019

AGRADECIMIENTOS

Este proyecto supone el final a una etapa intensa y de muchas emociones que, a pesar de las dificultades, siempre llevaré conmigo.

Quiero en primer lugar agradecer a mi tutor, Jose Antonio, la ayuda, el criterio, y el constante ánimo que me ha dado durante el desarrollo del trabajo, que volvería a repetir con él si tuviese otra oportunidad.

También dar las gracias a mis compañeros de clase y amigos por todas las experiencias que hemos compartido en estos años de carrera que siempre recordaré.

Por último, a mis padres, por su apoyo, sacrificio y consejo durante todo este duro viaje que nos ha tocado vivir, y a mi hermana Marta, que desde donde esté nos guía y ayuda a ser mejores personas cada día.

ABSTRACT

In an increasingly automated world where new technologies allow almost anything imaginable to be done, companies are discovering new ways of knowing what users think about their brand and new products they bring to market. Social networks are today the megaphone of society, and companies know that from these, very valuable information and directly related to them can be obtained. That's the reason why 91% of retail brands use two or more social channels, or 81% of small and medium businesses use some kind of social platform.

One of the technologies used is the one derived from Natural Language Processing (NLP), which allows understanding language, recognising speech, generating languages or analysing the sentiment of a text. During this work the focus will be on this last functionality, since it gives the possibility of perceiving positivity, negativity, irony, uncertainty, indecision... based on the text published by a person.

The use given to it is something that has been the subject of debate in recent years, since in some cases it can be considered unethical, as in politics. With programmes based on NLP you can know what people think about different issues, and on that basis change the political discourse. For reasons like this, social networks like Twitter ask that applications developed using its API aim to maintain the quality of the conversation, avoiding programs that produce spam, manipulate conversations or invade the privacy of users.

In this end-of-career project, a platform that allows users to search for any topic and get the sentiment of the responses that come from Twitter related to the search, has been developed. These answers can be up to a week old from the moment they are consulted or obtained in real time. In addition, the information is stored in a server-side database. On the client side you will be able to visualize different graphs with interesting data, such as the number of tweets per feeling or the place from where it is published. Finally, all data can be downloaded in JSON format to work with them later.

The aim of this work is to demonstrate the usefulness of this type of application and the savings it can produce for companies that spend large amounts of resources on actions and campaigns from which the desired effect does not arise. It also wants to show how easy it can be for organizations, discover what their users say on the Internet and adapt to new demands.

Introduction and motivation

People's opinions have always been a source of highly valued information for companies. From the smallest to the largest, they want to know what their users think, so they can make the decisions needed to ensure the success of their business. Thanks to this information, they fix bugs that may exist in products that are already being marketed and develop improvements for future versions. Even in politics similar situations can be found, where parties make use of tools that allow them to know the public impact of what they say, measuring and adjusting their speeches.

We live in a computerized era, in which 1 Exabyte (10^18) is produced every day, and in which every two days more data is produced than all the generated until 2003. A moment in which everything that surrounds people is capable of obtaining, creating and transforming information and, consequently, their way of working and relating.
Therefore, if before the problem was getting the information, now it is to manage it. Those organizations that have the necessary tools to analyze large amounts of information will undoubtedly survive in today's market. The challenge is to find the necessary data, in a structured, useful way, that allow to lead to solutions that really benefits. That is why this thesis is relevant, since it focuses on one of the most interesting subjects by companies, sentiment analysis. There is no doubt that the analysis of opinion is something current, and this is demonstrated by the numerous talks, research, platforms and studies that we can find today.

Thanks to this technique, the project comes to life, since it allows from the collection of large batches of tweets, the possibility to analyze and observe if what users say about a product or brand is positive, negative or neutral. Would it be possible that nowadays a company does not have tools that would give them the opportunity to see what their users think? The reality is that they don't, and this project seeks to demonstrate how extremely useful it can be to have them.

To end this section, it should be noted that the analysis is somewhat complex and will never be 100% accurate. Many factors must be considered, such as context, ambiguity, sarcasm or variations that may exist in the same language within different regions. Even humans sometimes find it difficult to agree on whether an opinion expresses one thing or another. The sophistication of these algorithms is something that is actively worked on and that will continue to improve, including all the tools that use them.

Goals and objectives

The objective of this project is to design, implement and evaluate a platform capable of simulating what could well be a 'Social listening' tool in the market. This web application will not only be able to analyse any text, which is the motivation discussed in the previous section. We want to have two basic functionalities related to the moment in which the data was published. They would therefore be the analysis in real time, or search based, which are data that were published a week before the search at the latest. When we talk about data we are referring to tweets, which is the information we are going to work with as it is a social networks listening App. Twitter is developed so that users can express their opinion in not too many characters, but as many times as they want. We have more information than anywhere else, which makes it perfect for these circumstances.

Back to the platform, in the analysis in real time section, the user will be able to search for any topic of interest and see in a graph how the data is entering with its corresponding feeling. The data obtained will be in English due to the sentiment analysis tool we use. At the end of the analysis, graphs will be displayed with information that may be relevant to the user.

As far as search-based analysis is concerned, we have the same graph as in the previous functionality, but we intend to add graphs that show the origin of the data, more positive

or negative publications, and number of tweets per feeling, which can be positive, negative or neutral.

It will also be possible to collect as many tweets as the user wants and will have the option of downloading the file in JSON to work with the data obtained locally. On the server side we also want to have these data in case they could be of interest.
The platform will be developed in Python programming language, currently the most popular. This choice is made because, after a first phase of analysis, many of the libraries that are believed to be useful for achieving the objective are designed for this language, to which must also be added its ease of use and large user base.

Finally, we want to give the user experience the importance it deserves. The objective is that the user can see the results in real time without affecting the experience, so the graphics should be updated without having to refresh the web application.

State of art

By doing some research in several articles and websites all over the internet, we conclude that natural language processing still has a long way to go, and many industries will benefit from this technology. In the sentiment análisis, a discipline within the NLP, there are already many applications. Here are some of them:

- **Voice of customer (VoC)**: Sentiment analysis is useful in understanding Voice of Customer because it helps to understand issues customers are experiencing over time, along with the reason. This empower internal teams by giving them a deeper view of this experience the customer is having.

- **Social media monitoring**: By using sentiment analysis on social media, we can get useful insights about dialogs that are happening around a label.

- **Market research and analysis**: Sentiment analysis empowers all kinds of market research. Whether exploring a new market, exploring what could be future trends, or keeping track of the competitors, this technology can make all the difference.

- **Voice of employee**: In the same way we measure VoC via customer surveys, we can do the same thing with employees and act on feedback received. Employees will give more actionables ideas on how to improve their workplace and be more productive.

- **Product analytics**: Whether by analyzing surveys, customer feedback received, or social media, sentiment analysis combined with machine learning enables big quantities of product feedback in a single moment.

After seeing the different functionalities that sentiment analysis presents, we proceed to research tools that exist in the market that offer similar solutions to ours. The following platforms share the same goals as the platform developed for this project, MRT.

- **Repustate**.
  Repustate is a natural language processing-based company. It has an API dedicated exclusively to the analysis of feelings available in several languages.

- **Monkeylearn**
  MonkeyLearn is a platform that through its API allows to receive data in text form to analyze and classify data. The company offers different solutions:
  - News classification. It allows to categorize a news in function of the text written in it.
  - Sentiment analysis. Detecting opinions in texts.
  - Abuse detection. Very useful in cases of bullying in social networks.

- **Block six analytics**
  The company has a Social Sentiment Analysis Platform (SAP), which uses machine learning and natural language processing to determine brand interest and value.

- **Meaningcloud**
  Meaningcloud is a Spanish company that defines its platform as "a simpler, more powerful and affordable way to extract the meaning of all types of unstructured content: social conversations, articles or dossiers".

In addition to the final platforms cited above, there are other tools that are directly related to sentiment analysis and that allow natural language processing applications to be developed from scratch without relying on a third-party API. This is the case of Natural Language Toolkit (NTLK), an open source library with which to develop applications in the Python programming language of natural language processing.

Implementation

The system has different components, where each one has different functionalities, constituting MRT:

- **Communication with the App web**: In this component are all the mechanisms that allow the collection of data through the application and the subsequent sending with the results. It is the communication channel between both sides, except for a small feature of the real-time analysis component.

  For the Web application we make use of Flask, a micro-framework based on Python language that allows the development of applications in a simple way with a minimum number of lines of code. It is considered micro because, although it is true that it is very easy to develop an app, it does not have too many extensions, which sometimes can limit. That is why the extension "Flask-Bootstrap" is used for the Frontend of the App. In addition, Flask includes a development Web

server, so no other infrastructure is needed to run the code, so it's easy to see what you get. In addition, it has a debugger that points out in case of error where it may be taking place. This makes it easier to have the code not very dispersed in different places.

Another important aspect is communication. In the application, client-server communication is the most important point. We need it to be as fast and smooth as possible, without sudden jumps or interruptions. In the case of MRT, the HTTP protocol is very limited, because in real time analysis it is necessary that the tweet display graph is constantly being updated without the need for the client to make a request. Through the HTTP protocol, the application would not be possible, because the page would be updated by not letting the user seeing what is happening. In this situation it would not only be a worse customer experience, but it would be impossible to develop the functionality, and therefore there would be no experience at all. To avoid this, websockets are used. Websockets allow us to transmit information in both directions at the same time as events occur. It is not limited to the request of one and response of the other. It is a bidirectional communication between both.

- **Search**: Component in charge of collecting API information, analyzing it, storing it and sending it back. It contacts all the modules of the system except for the Web application.
  The component that conforms the search performs several functions that will be briefly explained as follows:

  - **Check that the Tweet is not in the DDBB.**
    This will be important when running several searches on the same topic, because if not too much time has passed and the topic is not commented frequently, we can find repeated tweets that could be a problem. In the analysis this situation can never occur because the tweets are being published while the program is running.

  - **Extract useful information from the Tweet:**
    Tweets are JSON objects that have many fields. You would only have to extract those that are considered important, which are the following:
    - **Id_str**: String representation of the unique identifier of the Tweet.
    - **'User' -> 'Name':** Username of the person who publishes the content.
    - **'Created_at':** UTC time of the published Tweet. As it is the UTC time zone, we need to transform it to CEST.
    - **'Full_text':** Many of the Tweets today exceed the old limit of 140 characters, so we need the field where the full text is.

It would only be necessary the text, since from there the feeling is obtained, but it was important to provide it with veracity so that the user could at any time find the Tweet if desired and see that it exists. In addition, the identifier will be the primary key in the database, as it is unique and unrepeatable.

- **Analyze text:**
  To explain how the text is analyzed we have to explain how VaderSentiment works, which is the open source library that we use, and from which we get a value of between -1 and 1. This value tells us how very negative or positive has been the text analyzed. The text used is previously cleaned. Links, user mentions, and special characters are removed.

  VADER is based on a dictionary that assigns lexical characteristics to emotional intensities called feeling scores. The feeling score of a text can be obtained by summarizing the intensity of each word in the text. In a typical tweet, we can find not only words, but also emoticons and colloquialisms which are also assigned decisive values.

  Contextual elements, like punctuation, capitalization, and modifiers, also give feeling. VADER's sentiment analysis considers them when we talk about heuristics. The effect of these heuristics is quantified using human evaluators. Here are some of them:

  - **Punctuation**: Let's compare 'I like it' and 'I like it!'. The second has more intensity than the first, and therefore the feeling score has to be higher. VADER has this. To do this, he first calculates the sentence score without paying attention to the exclamations. Once we get a value, subtract or add depending on the previous result.

  - **Capitalization**: Lets now compare 'An INCREDIBLE performance' and 'an incredible performance'. The second one has definitely more intensity, and VADER takes it into account, adding or subtracting depending on the final result.

  - **Polarity change due to 'but'**: Often, 'but' connects two parts of a same sentence with contrasted feelings. The feeling, however, always tends to be the last. For example, 'I liked the film, but I don't want to see it again'. The first clause 'I liked the film' is positive, but the second 'but I don't want to see it again' is negative. It is obvious that the second one is the one that prevails. VADER reduces the value of the first clause by 50% and increases the second by 150%.

These are some of the heuristics used by VADER. Therefore, the value obtained is the result of combining a dictionary of lexical characteristics with the set of heuristics. The model normally works better when applied to social network texts, such as those used in the project, than large texts such as opinion articles that can be found in newspapers.

- **Insert Tweet in the database:**
  Pymongo, a library that connects the application developed in Python with the DDBB, is used for this. The fact of using a non-relational database is due, among other things, to the fact that a connection between collections was not necessary. Each of them is a topic searched by the user, and their respective documents, the tweets obtained. They were also looking for a data storage technology that was easy to implement and fast. Documents are JSON objects, which within MongoDB are known as BSON, binary JSON. Therefore, after knowing how we are going to store our information, we extract it. Next, we analyze the text, and a new field called 'Sentiment' is created to store the feeling obtained. After this, it is saved in the DDBB. This process will be repeated as many times as Tweets have been requested.

- **Prepare data to be transmitted by the Websocket:**
  Once we have performed the tweet analysis and saved it in the database, we leave all the data ready to be passed through Websocket when it is communicated to us. These data are the number of total tweets, number of original tweets of the total, number of retweets of the total, the average feeling of all of them and the date of the oldest and the most recent. It should be noted that sometimes the number of tweets specified by the user is not collected, and this can be seen in the data provided. After multiple tests it has been seen that the problem does not reside in our application and may be due to Twython, the library with which we collect information, or with the API itself Twitter search.

- **Real-time analysis**: Component of the MRT system that extracts the tweets at the same time they are published. As in the previous component, it is responsible for analyzing, storing in the database and returning the result for the user's visualization

  In contrast to search component, where data had already been obtained before connecting server and client, it is not the same here. Once the client message is received specifying that it is a real-time analysis, a process very similar to the search is executed. Extraction of information, obtaining of feeling, insertion of Tweet and creation of structures. When the number of tweets requested by the

user has been reached, the method that connects to the Twitter relay API will disconnect it and the necessary structures for creating the graphs will be transmitted through the Websocket. Thanks to the speed of the socket, this will be done in a matter of tenths of a second from the moment the retransmission ends, without affecting the user experience.

Results and evaluation

At the end of the implementation period, different tests are carried out to verify the correct functioning of the platform. These tests will be described together with the result obtained:

- **Execution of the code that connects to both Twitter Search API and Streaming API. Search and streaming of tweets will be tested from the integrated develpment enviroment.** Tweets are shown per console with all the specified fields

- **Execution of the code that deploys the web application.** We enter the local server path and see that the style sheets have been loaded, as well as seeing that a connection message has been printed per console.

- **Execution of different searches and analysis in real time while the system is running.** We make use of the different functionalities with total normality, visualizing the results.

- **Download the JSON document in the results page.** After executing a search, a JSON document appears that contains the data obtained and that coincide with the data stored in the database.

- **Attemp search or analysis without filling all the required fields.** When trying to run a search or analysis without filling in all the fields that are requested, a box appears in the empty field demanding to be filled in.

As shown, the system proves to be successful. All information is obtained in English.

Economic and social impact

The main objective of the project is that companies find in MRT a tool with which to better understand their clientele and what steps should be taken to improve their business. At a time when digitization is the priority for companies in all sectors, MRT offers a cheaper and more accurate solution than many of the campaigns and projects that could have been carried out that had the same objective. Therefore, the socioeconomic impact sought with this end-of-career work is the improvement in the use of resources by

companies, which logically think constantly how to improve their products and services taking into account the opinion of both their most loyal customers and those who do not use them so often. This saving can ultimately mean the creation of more jobs by the company, or even wage and job improvements. In addition, many organizations could originate thanks to these new tools, becoming consultants specialized in data processing. Another aspect to bear in mind is that of customers. Many times, they feel that companies do not listen to their complaints and that they do not put land in the way to improve their products and services. Thanks to MRT, users can find in it a loudspeaker where they can be sure that their requests will at least be heard.

Not all solutions can be aimed at the private sector. Also, public sector can see MRT as an instrument with which to better understand the needs of the inhabitants and, therefore, facilitate and improve their lives.

MRT is a versatile tool that with not much complexity can be adapted to any written input that is considered and therefore has a great potential that can be even greater, as there is much room for improvement. As in almost everything else, the system can be used for an unethical purpose, which is that related to politics. The commitment that MRT is not used for this is total and can only be used with an approval after a previous process of verification of the activities of the organization.

Conclusion

The first conclusion that is obtained, and which is the most important, is that the objectives that were set for the present work have been met successfully after the completion of it.

It has been possible to build a system that, by means of a web application, gives us information about what users think in a social network, Twitter, regarding a sought-after theme. In addition, the user of our platform is given complete freedom to work with the data obtained in future research on the subject sought. Another objective that has been achieved is related to the design of the application. From the beginning, the user experience has been the central axis of the project, which at no time has been neglected, becoming a priority. It is considered that in this area the objective has been successfully fulfilled. It has a clean, minimalist design, but without lacking functionality, which attempts within the complexity of it, to facilitate the viewing of data to the user. One aspect also to bear in mind and related to the user experience with which many doubts arose, was the possibility of updating the graphics without having to constantly recharge the application. After researching and finding the necessary technologies, the objective was more than fulfilled, allowing the client to see the data obtained in real time.

The system has a very low execution time for the amount of data that is worked with, responding quickly and efficiently to user requests that are generated. Therefore, it could be considered to continue working with the system to improve it, as there are many things to implement and improve, which will be discussed in the next section.

Finally, I would like to highlight the theme chosen for this final thesis. This is an interesting project, based on the demand that exists today around the data treaty, and that meets the expectations with what is considered a dissertation in the degree of computer engineering.

Future Works

The analysis will never be accurate if it is not considered the context surrounding a tweet. It would not be difficult to incorporate another sentiment analysis tool if it proves to be better than the current one, VaderSentiment, so future developers should take this aspect into account, based in the context fact.

As far as data visualization is concerned, more graphs could be incorporated to explain the results obtained. It was contemplated but not implemented a tool to show the most repeated words. With more time it would have been one more functionality in the system, and it will be undoubtedly added in the future. Other graphs, such as a world map showing where the tweets are being received from, or the possibility of seeing each tweet that enters the analysis in real time, could be contemplated, improving the user experience and squeezing all the data they offer us from Twitter. There are many limitations becuase of not making use of their payment version API. If a company were very interested in the tool, it would be suggested to buy access to the "API for companies", with which you have access to all the tweets since the social network exists and therefore the searches would be much more significant.

We also want to highlight another important improvement, which is the languages. The project has only been developed for information found in English, but if organizations that work with different languages find in MRT a tool with which to obtain data, the system should be adapted to be able to be used with the new one.

Finally, the possibility of having an accesible application for everyone. Currently it can only be used locally, but with tools such as Google App Engine could be considered to launch it so that anyone can use it responsibly.

As it can be seen, MRT has many improvements that would imply having a business strategy, as each one of them directs us towards a more private or public sphere. The system has a lot of potential and there is no doubt that it will be considered for further improvement soon.

RESUMEN

En un mundo cada vez más automatizado y en el que las nuevas tecnologías permiten hacer casi cualquier cosa imaginable, las empresas están descubriendo nuevas formas de conocer lo que los usuarios piensan sobre su marca y nuevos productos que lanzan al mercado. Las redes sociales son hoy en día el megáfono de la sociedad, y las empresas saben que a partir de estas se puede sacar información muy valiosa y directamente relacionada con ellas. Es por ello por lo que el 91% de las marcas retail utilizan dos o más canales sociales, o que el 81% de los pequeños y medianos negocios utilizan algún tipo de plataforma social [1].

Una de las tecnologías que se utilizan es aquella que deriva del Procesamiento del Lenguaje Natural o "*Natural Language Processing*" (NLP), que permite comprender el lenguaje, reconocer el habla, generar lenguajes o analizar el sentimiento de un texto. Durante este trabajo se pondrá el foco sobre esta última funcionalidad, ya que da la posibilidad de percibir positividad, negatividad, ironía, incertidumbre, indecisión… basado en el texto publicado por una persona.

El uso que se le da es algo que está siendo objeto de debate en los últimos años, ya que en algunos casos se puede considerar poco ético, como en la política. Con programas basados en NLP se es capaz de saber lo que piensa la gente sobre distintos temas, y en base a eso modificar el discurso político. Por razones como esta, redes sociales como Twitter piden que las aplicaciones desarrolladas que hagan uso de su API tengan como objetivo mantener la calidad de la conversación, evitando programas que produzcan spam, manipulen conversaciones o invadan la privacidad de los usuarios [2].

En el presente proyecto de fin de carrera se ha desarrollado una plataforma que permite a los usuarios buscar cualquier tema y obtener el sentimiento de las respuestas que provienen de Twitter relacionadas con la búsqueda. Estas respuestas pueden tener hasta una semana de antigüedad desde el momento que se consulta u obtenerse en tiempo real. Además, la información queda almacenada en una base de datos del lado del servidor. Del lado del cliente se será capaz de visualizar diferentes gráficas con datos de interés, como el número de tweets por sentimiento o el lugar desde donde se publica. Por último, se podrán descargar todos los datos en formato JSON para trabajar posteriormente con ellos.

Con este trabajo se quiere demostrar la utilidad que tienen este tipo de aplicaciones y el ahorro que puede suponer para las empresas que se gastan grandes cantidades de recursos en acciones y campañas de las que no surge el efecto deseado. También se quiere mostrar lo fácil que puede ser para las organizaciones, descubrir qué opinan sus usuarios en internet y adaptarse a nuevas demandas.

Palabras clave – Redes sociales, NLP, Twitter, base de datos, servidor, cliente, JSON, plataforma, sentimientos, búsqueda, análisis en tiempo real, API.

CONTENIDO

Índice de ilustraciones

Índice de tablas

Capítulo 1: Introducción

1.1 Motivación del proyecto

La opinión de las personas siempre ha supuesto una fuente de información muy preciada por las empresas. Desde las más pequeñas a las más grandes, se quiere conocer qué piensan sus usuarios para poder tomar las decisiones necesarias y garantizar el éxito de su negocio. Gracias a ellas, las opiniones, arreglan errores que puedan existir en los productos que ya están siendo comercializados y desarrollan mejoras para futuras versiones. Incluso en política pueden encontrarse situaciones similares, donde los partidos hacen uso de herramientas que les permite conocer el impacto público de lo que dicen, midiendo y ajustando sus discursos [3] [4].

Se vive en una era informatizada, en la que se produce al día 1 Exabyte (10^18), y en la que cada dos días se producen más datos de los generados en conjunto hasta 2003 [5]. Una época en el que todo lo que rodea a las personas es capaz de obtener, crear y transformar información, y, en consecuencia, su forma de trabajar y relacionarse.

Por tanto, si antes el problema era conseguir información, ahora es manejarla. Aquellas organizaciones que posean las herramientas necesarias para analizar grandes cantidades de datos serán las que indudablemente sobrevivan en el mercado actual. El reto está en conseguir la información necesaria, de forma estructurada, útil, y que permitan llevar a soluciones que realmente supongan un beneficio. Es por eso por lo que el presente trabajo de fin de carrera cobra especial relevancia, ya que pone el foco en una de las materias más interesantes y buscadas por las empresas, el Procesamiento del Lenguaje Natural (NLP, por sus siglas en inglés *Natural Language Processing*). No cabe duda de que el análisis de la opinión es algo actual, y así lo demuestran las numerosas charlas, investigaciones, plataformas y estudios que se pueden encontrar hoy en día. En España, uno de los estudios más importantes los lleva a cabo Deloitte Digital, que presenta cada año un Estudio Nacional sobre el Nivel de Operativización de Experiencia de Cliente en el mercado español [6]. En éste, se evalúa el trabajo que se está llevando a cabo en muchas de las empresas más importantes del país en materia de experiencia de cliente. Esto es algo directamente relacionado con el análisis de sentimiento, pues gracias a ello permite a las empresas actuar en función del *feedback* de sus clientes. La experiencia de cliente se detallará de forma más extensa en el *Capítulo 2: Estado del Arte*.

Gracias al NLP, el proyecto cobra vida, ya que permite a partir de la recolección de grandes lotes de tweets, datos para trabajar, analizar y observar si lo que dicen los usuarios sobre un producto o marca es positivo, negativo o neutro. ¿Sería posible que alguna empresa en la actualidad no contase con herramientas que les diese la oportunidad de conocer que piensan sus usuarios? La realidad es que no, y este proyecto busca demostrar lo sumamente útil que puede ser contar con ellas.

Para finalizar este apartado, cabe destacar que el análisis es algo complejo y que nunca será 100% exacto. Hay que tener en cuenta muchos factores, como el contexto, la ambigüedad, el sarcasmo, o variaciones que pueden existir de una misma lengua en diferentes regiones. Incluso a los humanos nos cuesta en ocasiones coincidir en si una opinión expresa una cosa u otra. La sofisticación de estos algoritmos es algo en lo que se trabaja de forma activa y que seguirán mejorando junto a todas las herramientas que los utilizan. De esto se hablará con más detalle en el *Capítulo 2: Estado del Arte*.

1.2 Objetivo del proyecto

El objetivo de este proyecto es diseñar, implementar y evaluar una plataforma capaz de simular lo que bien podría ser una herramienta de '*Social listening*' en el mercado. Esta aplicación web no sólo podrá analizar un texto cualquiera, que es la motivación de la que se ha hablado en el apartado anterior, sino que se quiere contar con dos funcionalidades básicas relacionadas con el momento en el que el dato fue publicado. Actualmente, la sociedad quiere respuestas a sus demandas, en ocasiones, en el mismo momento en el que se realiza una acción, y en otras, tras un tiempo un dado. Teniendo en cuenta esto, se quiere desarrollar una herramienta que analice en tiempo real, pero que también permita la búsqueda de datos que fueron publicados previamente, en concreto, una semana antes como muy tarde. En este contexto, debemos tener en cuenta que hablar de datos es hablar de tweets, que es con lo que se va a trabajar al tratarse de una App de escucha de redes sociales. Twitter está desarrollada para que los usuarios puedan expresar su opinión en no demasiados caracteres, pero tantas veces como se quiera. Se cuenta con más información que en cualquier otro blog personal, lo que lo hace perfecto para estas circunstancias.

Volviendo a la plataforma a desarrollar, en la parte de análisis en tiempo real, se permitirá la búsqueda de cualquier tema que fuera de interés y ver en una gráfica cómo se actualizan los datos con su correspondiente sentimiento. Los datos obtenidos serán en inglés debido a la herramienta de análisis de sentimientos que se utiliza. Al finalizar el análisis, se desplegarán gráficas con información que pudiera ser relevante para el usuario.

En lo que al análisis basado en búsqueda se refiere, se quiere contar con las mismas gráficas que en la funcionalidad anterior, que muestran la procedencia de los datos o número de tweets por sentimiento, pudiendo ser positivo, negativo o neutral. También se incorporará un panel que muestre la publicación más positiva y negativa de la búsqueda. Además, será posible recopilar tantos tweets como el usuario quiera, y se contará con la opción de descargar el archivo en JSON para poder trabajar con los datos obtenidos en local. Del lado del servidor también se quiere contar con estos datos por si pudiesen ser de interés.

La plataforma será desarrollada en el lenguaje de programación Python, actualmente el más popular [7]. Esta elección se lleva a cabo debido a que, tras una primera fase de análisis, muchas de las librerías que se cree que pueden ser útiles para la consecución del

objetivo están diseñadas para este lenguaje, a lo que se debe añadir también su facilidad de uso y gran base de usuarios.

Por último, se quiere dar la importancia que se merece a la experiencia de usuario. El objetivo es que este pueda visionar los resultados en tiempo real sin que la experiencia se vea afectada, por lo que las gráficas deberán actualizarse sin que haya que refrescar la aplicación web, pues de esta última forma sería imposible una cómoda visualización.

1.3 Entorno operacional

En este apartado se exponen las herramientas que han sido utilizadas a lo largo del desarrollo del proyecto, tanto en software como en hardware.

La herramienta hardware principal ha sido el ordenador personal, que cuenta entre otras con las siguientes especificaciones:

- **Procesador** Intel® Core™ i7-6700HQ (6M Cache, 2.6GHz hasta 3.5GHz)
- **Memoria RAM** 8GB DDR4 SODIMM
- **Sistema operativo** Windows 10 Home 64 bits

Las herramientas software utilizadas han sido:

- Paquete ofimático de Microsoft Office Standard 2016:
  - Microsoft Word 2016
  - Microsoft Excel 2016
- Microsoft Project Standard.
- Entorno de desarrollo integrado (IDE), PyCharm [8].
- Editor de texto y de código fuente libre, Sublime Text 3 [9].

1.4 Contenido de la memoria

El presente documento constituye la memoria del trabajo de fin de carrera, que recoge todo el trabajo que se ha llevado a cabo sobre el mismo. La estructura del documento es la que se presenta a continuación.

En el Capítulo 1 se realiza una introducción, en la que se incluye la motivación que impulsa el desarrollo del proyecto, el objetivo que se persigue y el entorno operacional sobre el cual se realiza el trabajo. Por último, se expone cómo va a ser organizado el contenido del documento, y una serie de acrónimos y definiciones que pueden ser relevantes para la comprensión del presente documento.

En el Capítulo 2 se lleva a cabo un análisis sobre las diferentes herramientas que existen en el mercado que comparten un objetivo similar a nuestro trabajo de fin de carrera, además de un previo estudio de la situación actual de la tecnología que se utiliza.

En el Capítulo 3 se describe la etapa de análisis, en la que se establecen las funcionalidades del sistema y cómo van a interactuar los diferentes actores con ellas.

En el Capítulo 4 se expone el diseño del sistema, es decir, cómo van a interactuar los distintos elementos que lo componen entre ellos y en qué orden. Además, se describirá brevemente su funcionalidad para que el lector tenga una ligera de idea de su tarea.

En el capítulo 5 se describe de forma detallada el proceso llevado a cabo para la implementación del sistema, explicando también las diferentes herramientas que se han integrado en el proyecto.

En el capítulo 6 se muestran las distintas pruebas unitarias del sistema, así como el despliegue de una matriz de trazabilidad que muestra cómo los requisitos establecidos han ido siendo verificados uno a uno.

En el Capítulo 7 se describen aspectos relacionados con la gestión del proyecto, como la planificación, el presupuesto y su impacto socioeconómico.

En el Capítulo 8 se relatan las conclusiones técnicas y personales que se han obtenido durante la realización del trabajo de fin de carrera. También se incluyen posibles trabajos futuros que pueden llevarse a cabo con el sistema.

Por último, se indican las referencias y bibliografías consultadas para el desarrollo del sistema que compone el proyecto.

1.5 Acrónimos

IA - Inteligencia artificial

NLP – Natural Language Processing

JSON – Javascript Object Notation

APP – Application

ML – Machine Learning

BBDD – Base de datos

CV – Curriculum Vitae

HTML – Hypertext Markup Language

1.6 Definiciones

Scraper: El web scraping es el proceso de extracción de datos de sitios web. Todo el trabajo se realiza mediante un código que se denomina *Scraper*. Primero, envía una

consulta "GET" a un sitio web específico. Luego, analiza un documento HTML basado en el resultado recibido. Una vez hecho esto, el *Scraper* busca los datos que necesita dentro del documento y, finalmente, se obtienen.

Toolkit: Set de herramientas software.

Framework: Un framework proporciona código que establece una aplicación completa ideal que sabe "qué" hacer, pero no sabe "cómo" hacerlo. Las acciones específicas (el "cómo") las lleva a cabo el programador.

Social listening: Proceso de monitorización de conversaciones digitales para entender lo que los clientes están diciendo sobre una marca y/o una industria en tiempo real.

Capítulo 2: Estado del arte

En este capítulo se expone la situación actual de aquellas tecnologías que se encuentran relacionadas con el proyecto de fin de carrera. Entre ellas, se encuentra el análisis de sentimientos, que es una de las disciplinas que componen el Procesamiento de Lenguaje Natural (NLP). Se detallarán los diferentes casos de uso que tiene en el mercado, algunas de las herramientas comerciales que comparten el mismo objetivo que nuestro proyecto y retos futuros de los equipos que trabajan en esta disciplina.

2.1 Procesamiento del Lenguaje Natural

El Procesamiento del Lenguaje Natural es un término utilizado para referirse a la capacidad que tienen las máquinas para procesar y comprender el lenguaje tal como está escrito o hablado por los seres humanos. Es una forma de IA que trata de analizar y entender el lenguaje en el contexto en el que está siendo utilizado. Un ejemplo básico sería el texto predictivo que aparece en los teclados al escribir, en el que el sistema intenta adivinar continuamente cuál va a ser la siguiente palabra que va a ser utilizada. Pero esta no es la única aplicación que tiene el NLP. A continuación, se describen casos de uso de la actualidad.

- **Análisis de sentimientos**: Las organizaciones están buscando nuevas formas de conocer a sus clientes con el que servirles de una manera personalizada. Usando el análisis de sentimientos se puede determinar aquellos que se encuentran detrás de las palabras, dando la capacidad de conocer el comportamiento del cliente. Esto se convierte pues en un factor crucial en la toma de decisiones de una empresa. El presente proyecto de fin de carrera hará uso de esta rama del NLP, y por tanto se explicará con más detenimiento en este mismo capítulo.

- **Chatbots o sistemas conversacionales:** El NLP permite que los chatbots, sistemas computacionales diseñados para proveer información o dar soporte a través de una conversación, entiendan los mensajes de clientes y respondan adecuadamente. El procesamiento del lenguaje también les ayuda a entender el contexto y el significado de términos utilizados para ofrecer una mejor respuesta. Esto será de gran utilidad en sectores como el de la medicina, que se detallará más tarde.

- **Extracción de contenido relevante**: En la época actual, la cantidad de información con la que se trabaja es ilimitada. Las empresas se ven obligadas a procesar grandes cantidades de documentos para rescatar información importante y que ofrecen valor al negocio.

- **Traducción automática a diferentes idiomas**: La traducción automática de nuestro idioma al del receptor sin la intervención de un intermediario es algo ya habitual y con lo que hace unos años no se contaba. Esta traducción aporta mayor productividad en proyectos y permite entender información en idiomas con los que nunca se ha estado en contacto.

- **Reconocimiento y síntesis del habla**: Siri de Apple, Alexa de Amazon o Aura de Telefónica son ejemplos de casos de uso de esta tecnología, capaz de reconocer lo que le decimos a nuestro dispositivo y dándonos la mayoría de las veces la solución.

Estas son algunas de las aplicaciones que derivan del Procesamiento del Lenguaje Natural, y en las que se basan las empresas para el desarrollo de nuevas soluciones en este ámbito. A continuación, se procede a detallar cómo el NLP ha cambiado o está cambiando la forma en la que se trabaja en distintos sectores laborales.

- **Salud**: La medicina se ha convertido en un mercado atractivo para empresas que desarrollan chatbots, que se encuentran en las primeras fases de implementación. La firma de investigación de mercados Grand View Research, estima que para 2025, el mercado de estos sistemas alcanzará un poco más de los mil millones de dólares [10]. Estos programas utilizan algoritmos entrenados con grandes volúmenes de información de contenido clínico, como protocolos médicos e información de enfermedades crónicas, que ayudan a interpretar los síntomas de los pacientes y recomendar un diagnóstico adecuado. Para esto, se debe establecer una conversación con el paciente en el que sistema tiene que entender y hacerse entender, lugar donde NLP entra en escena. En la *Ilustración 1* se puede observar el proceso descrito.

  No todas las aplicaciones en el mundo sanitario basadas en NLP están relacionadas con Chatbots. En un artículo para *The Hospital Leader*, el blog oficial de *Society of Hospital Medicine*, una sociedad de membresía estadounidense para médicos y especialistas que se dedican a la medicina hospitalaria, el doctor Robert Watcher destaca el enorme potencial que tiene esta tecnología para la lectura de las notas que toma un médico al atender a un paciente [11]. En este, comparte una visión de futuro en la que el NLP es capaz de facilitar el trabajo a los profesionales sanitarios, liberándoles de rellenar plantillas y listas de control, permitiéndoles hablar más detenidamente con sus pacientes y describir sus hallazgos y pensamientos en prosa.

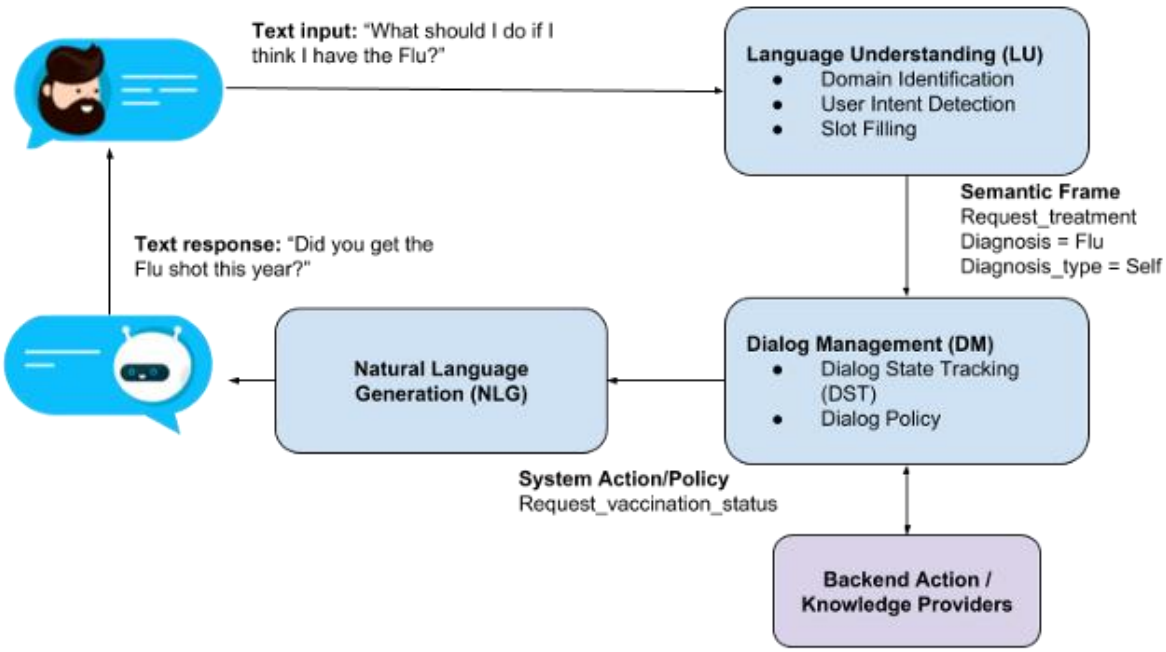

Ilustración 1: Chatbot en el ámbito de la medicina.
Fuente: https://medium.com/curai-tech/nlp-healthcare-understanding-the-language-of-medicine-e9917bbf49e7

- **Recursos humanos**: El NLP junto al *Machine Learning* (ML) está siendo utilizado en procesos de selección de candidatos para que antes de que un ser humano evalúe los CVs recibidos, los programas extraigan palabras clave previamente establecidas para decidir si una persona debe o no seguir en el proceso de selección [12].

- **Legal**: Los abogados ejecutan continuamente tareas monótonas cuando se preparan para un caso jurídico. Las soluciones basadas en NLP pueden extraer el significado de los documentos a una velocidad vertiginosa en comparación con la de un ser humano. Esto permite crear conexiones entre documentos mediante la identificación y relación de entidades relevantes. Los programas pueden examinar palabras individuales, su aplicación y contexto, y con qué frecuencia aparecen juntos, lo que permite identificar patrones en miles de registros. Esta tecnología puede ahorrar horas de trabajo, lo que significa que se ahorran horas facturables para más casos y clientes. Esto se aprecia en el documental "Vice: La mano de obra del futuro" [13], donde se muestra como un software basado en NLP, Lawgeex [14], supera a un abogado en la revisión de un acuerdo de confidencialidad. En este experimento, se calificaba en función de la velocidad y de la precisión. Tras el ejercicio, se comprobó que Lawgeex no era sólo más preciso formalizando el acuerdo y detectando errores, si no que tardaba menos de la mitad del tiempo que le tomó al abogado.

2.2 Análisis de sentimientos

2.2.1 Aspectos básicos

El análisis de sentimientos es una disciplina dentro del amplio mundo del NLP. Los programas que derivan de esta rama intentan identificar y extraer opiniones dentro de un texto, obteniendo su polaridad, es decir, la cantidad de positividad o negatividad, y el asunto, que es el tema sobre el que se opina.

Actualmente, el análisis de sentimientos genera más interés que nunca, puesto que cada vez hay más información, ya sea pública o privada, disponible en Internet, con gran cantidad de textos en foros, blogs y redes sociales. Con la ayuda de estos sistemas, esta información no estructurada podría transformarse en información de gran utilidad sobre productos, servicios, marcas, políticas, etcétera.

Se estima que el 80% de los datos mundiales no están estructurados, la mayor parte de redes sociales, encuestas y artículos [15]. Estos textos suelen ser difíciles y se requiere de muchos recursos para analizarlos y clasificarlos. Los sistemas de análisis permiten automatizar procesos, obtener información práctica y ahorrar horas de procesamiento manual de datos. Algunas de las ventajas incluyen:

- **Análisis en tiempo real**. Permite a las empresas encontrar clientes que estén a punto de darse de baja o utilizar sus servicios, y actuar ante tal situación. Esto genera dos procesos de trabajo. El primero es un ciclo interno en el que se identifican todas las acciones relacionadas con el servicio que se está ofreciendo. Se trata de un proceso de larga duración, pues al tratarse de un problema estructural hay que tomar las decisiones acertadas sin afectar a otras unidades de negocio. El segundo proceso sería la solución inmediata de los problemas del cliente, que tiene un margen de tiempo mucho más corto. Este primer proceso sólo ocurre si son muchos los casos que se repiten con el mismo problema. En la *Ilustración 2* se aprecia lo explicado.

- **Escalabilidad.** Permite clasificar una cantidad de información que de forma manual seria inimaginable. El análisis de sentimientos permite un procesado a escala de manera eficiente.

- **Criterio objetivo para información subjetiva.** El uso de un sistema de análisis de sentimientos permite que las empresas puedan aplicar los mismos criterios a todos sus datos, ya que los seres humanos no juzgan muchas veces de la misma manera una opinión. De hecho, solo se está de acuerdo entre el 60-65% de las veces [16]. Esto ayuda a reducir errores y mejorar la consistencia de los datos.

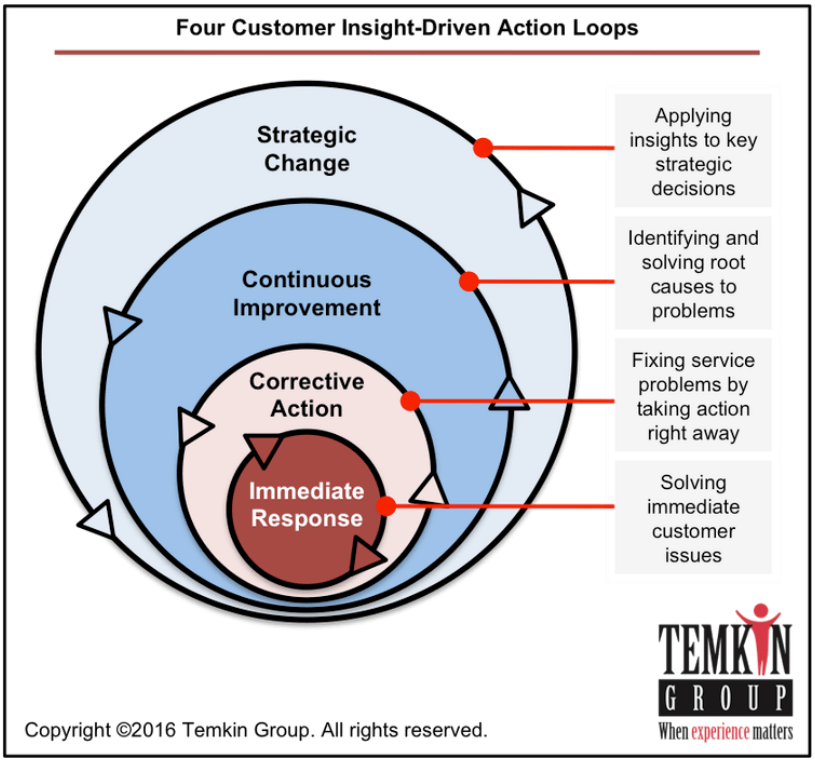

Ilustración 2: Procesos de trabajo en base al feedback obtenido. Fuente: https://experiencematters.blog/2016/08/15/use-customer-insights-to-close-four-loops/

2.3 Casos de uso en el mercado laboral

En este apartado se expone qué aplicaciones tiene el análisis de sentimientos en la vida real, y el impacto que ejercen sobre las empresas y la sociedad. Serán tratados pues los siguientes usos:

- Voz de cliente (VoC)
- Monitorización en redes sociales
- Investigación y análisis de mercado
- Análisis de voz de empleado
- Análisis de productos

2.3.1 Voz de cliente

Las empresas, a parte de la información que puedan encontrar en redes sociales, suelen enviar encuestas a sus usuarios, de las que se obtiene el *Net Promoter Score*, o NPS [17], que les ayuda a identificar a clientes promotores, pasivos o detractores. Esta clasificación está basada en la respuesta que se da a la pregunta *¿Recomendaría esta compañía/producto y/o servicio a un amigo o familiar? ¿Por qué?* La primera da la opción de poner una nota del 1 al 10, pero la segunda requiere de la introducción de texto.

La puntuación numérica sólo da un número, que sin un contexto del por qué esa nota, no sirve de demasiado. Es aquí donde el análisis del sentimiento se usa. Gracias a esto, la empresa es capaz de comprender matices de la experiencia del cliente, y ofrece a los integrantes de los equipos internos una visión más profunda. También permite, como se ha explicado anteriormente en el apartado de *Aspectos básicos,* dar una respuesta más rápida, lo que permite mantener la base clientelar.

Esto se puede apreciar con el siguiente ejemplo. La consultora estratégica *McKinsey & Company* [18] desarrolló una herramienta llamada *City Voices* [19] que consiste en la creación de encuestas con más de 150 indicadores diferentes, que, junto al análisis de sentimientos, permite a ciudades y países enteros a comprender cómo viven los ciudadanos y qué es lo que necesitan, invirtiendo en aquellos aspectos más descuidados. Este proyecto se implementó entre otros lugares en Brasil, que pudo hacer frentes a necesidades urgentes como la mejora de la seguridad en la red de autobuses.

2.3.2 Monitorización en redes sociales

Hoy en día las empresas tienen sus departamentos de mercadotecnia controlando la cantidad de menciones que recibe la compañía por redes sociales, ya que es algo importante. El número de menciones no tiene que significar que el producto o servicio esté siendo satisfactorio, también puede ser todo lo contrario. Por ejemplo, como ocurrió con el incidente de *United Airlines* en abril de 2017 [20], en el que se sacó a la fuerza a un pasajero por falta de asientos y fue grabado y publicado por uno de los que se encontraban en el avión. Sólo 24 horas después ya se había compartido la publicación más de 87,000 veces con casi 7 millones de reproducciones [21].

El análisis de sentimientos permite que las empresas, sin importar de su tamaño, ante situaciones donde su marca es tema de conversación, puedan entender el contexto e intervenir. También pueden aplicar esto en redes de empresas que ejerzan una competencia para evitar que les pueda ocurrir lo mismo en un futuro o para dar el paso que necesitan para liderar el sector.

Ilustración 3: Volumen de menciones United Airlines en Twitter, Facebook e Instagram.
Fuente: https://www.brandwatch.com/blog/react-united-airlines-overbooked/

2.3.3 Investigación y análisis de mercado

El análisis de sentimientos permite cualquier tipo de investigación de mercado. Esto significa que una empresa puede anticiparse a futuras tendencias, mejorando sus productos/servicios y obteniendo un elemento diferenciador sobre la competencia. Si a este proceso se le añade la automatización, generando informes relacionados con investigación de mercado, se estaría ante la herramienta definitiva para que las compañías cuenten con un repositorio relevante.

2.3.4 Análisis de voz de empleado

Al igual que se mide la voz del cliente para mejorar los productos y servicios una organización, lo mismo se puede hacer con la voz del empleado, actuando a partir de los comentarios de este. A diferencia de las opiniones que puede dar un cliente, en ocasiones destructivas, el empleado siempre va a expresar inquietudes sobre cómo mejorar el lugar de trabajo. Las personas que componen la alta dirección de una empresa siempre estarán interesadas en mantener a sus empleados comprometidos y capacitados para nuevos retos, y a través de herramientas de voz de empleado, este objetivo está a su alcance.

Ilustración 4: Voz de empleado
Fuente: https://www.mycustomer.com/experience/voice-of-the-customer/is-voice-of-the-employee-the-new-employee-engagement-or-a-whole-new

2.3.5 Análisis de productos

Hoy en día la única forma de que un producto tenga éxito es mediante la solicitud de opiniones de las personas que lo utilizan para poder seguir mejorándolo. Para muchos equipos, encontrar estos comentarios puede ser muy complicado, ya que no cuentan con una estructura que les permita obtenerlos, evaluarlos y clasificarlos. Es aquí donde el análisis de sentimientos entra nuevamente en juego. Mediante el análisis en encuestas, interacciones en la atención al cliente o redes sociales, se puede obtener toda la información necesaria, que, junto a un sistema de análisis y aprendizaje automático, puede evaluarse. Esto permitirá a las empresas mantener un seguimiento en lo que le gusta y lo

que no a sus clientes, atraer audiencias, y dotar de las herramientas necesarias a los equipos de desarrollo de producto que necesitan.

2.4 Herramientas comerciales relacionadas

A continuación, se analizarán distintas soluciones encontradas en el mercado que comparten parcialmente el objetivo del sistema desarrollado para el trabajo de fin de carrera:

- **Repustate** [22].
  Repustate es una empresa que se dedica al procesamiento del lenguaje natural. Posee una API dedicada en exclusiva al análisis de sentimientos disponible en varios idiomas.

- **Monkeylearn** [23].
  MonkeyLearn es una plataforma que a través de su API permite recibir datos en forma de texto para analizar y clasificar datos. La compañía ofrece diferentes soluciones entre las que se encuentran:
  - Clasificación de noticias. Permite categorizar la noticia en función del texto redactado.
  - Análisis de sentimiento. Detectar opiniones en textos
  - Detección de abusos. De gran utilidad en casos de acoso escolar en redes sociales

- **Block Six Analytics** [24].
  Esta compañía cuenta con una plataforma de análisis del sentimiento en redes sociales llamada *Social Sentiment Analysis Platform* (SAP), que utiliza machine learning y procesamiento del lenguaje natural para determinar el interés y el valor de las marcas.

- **Meaningcloud** [25].
  Meaningcloud es una compañía española que define su plataforma como "*una manera más sencilla, potente y asequible de extraer el significado de todo tipo de contenido no estructurado: conversaciones sociales, artículos o expedientes*" [25].

Además de las plataformas finales que se han citado, existen otras herramientas que están directamente relacionadas con el análisis de sentimiento y que permiten desarrollar aplicaciones de Procesamiento de Lenguaje Natural desde cero sin depender de una API de un tercero. Este es el caso de *Natural Language Toolkit* [26] (NTLK), una librería de

código abierto con el que poder desarrollar aplicaciones en el lenguaje de programación Python de procesamiento de lenguaje natural.

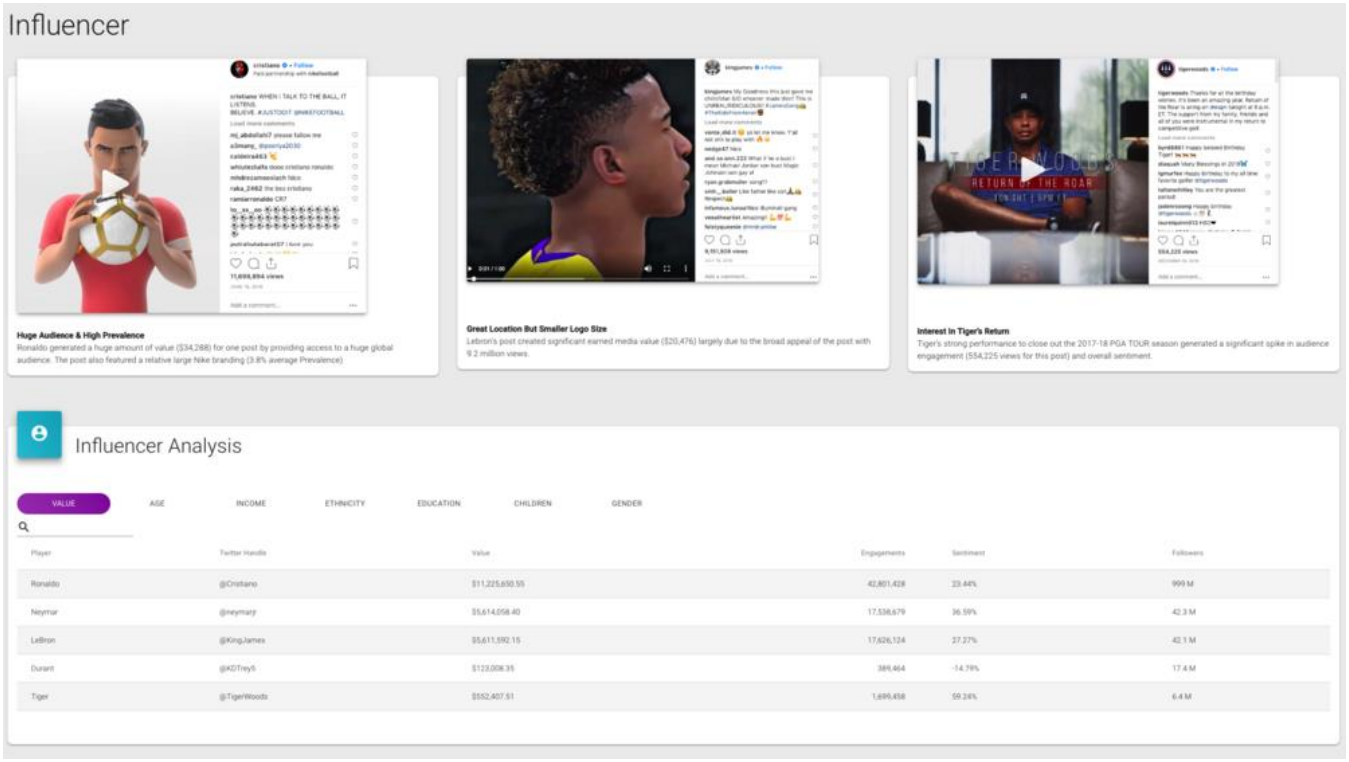

Ilustración 5: Plataforma Block Six Analytics.
Fuente: https://www.blocksixanalytics.com/social

2.5 Retos del análisis de sentimientos

La mayor parte del trabajo de equipos dedicados al análisis de sentimientos está relacionada con clasificadores de sentimientos más precisos, enfrentándose a los principales desafíos y limitaciones de este campo. A continuación, se explicarán algunos de los desafíos mencionados.

- **Subjetividad y tono.** La detección de la objetividad y subjetividad en textos es tan importante como el análisis de su tono. Se analizan los siguientes ejemplos: "*El reloj es bueno*", "*El reloj es plateado*". La mayoría de gente pensaría que el primero es más positivo que el segundo, que sería neutral. Por tanto, todos los predicados deben tratarse de manera diferente. En estos ejemplos, *bueno* es más subjetivo que *plateado*.

- **Sarcasmo e ironía.** La diferencia entre el significado literal y el deseado, es decir, la ironía, o el significado literal y su versión insultante, el sarcasmo, cambian el sentimiento de positivo a negativo y viceversa. La detección de la ironía y el sarcasmo requiere de un gran análisis del contexto en el que se encuentran los textos, y, por lo tanto, es de gran dificultad. Ante la respuesta a una pregunta con

un "*sí, claro*", se haría entender que el emisor está siendo positivo, pero al desconocer su contexto, podría estar refiriéndose a la situación contraria.

- **Contexto y polaridad.** Todo lo que se dice en algún momento, en algún lugar y para alguna persona, tiene una razón, es decir, que se pronuncia en un contexto. Intentar analizar el sentimiento sin el contexto es complicado. Uno de los problemas que surgen a partir del contexto es la polaridad. Se visualizan los siguientes ejemplos: "*¡Todo!*", "*Nada en absoluto*". Si la pregunta fuese "*¿Qué cosas te gustaron de la fiesta?*", la primera respuesta sería positiva y la segunda negativa. Ahora, si la pregunta fuera "*¿Qué cosas no te gustaron de la fiesta?*" el sentimiento de cada una sería completamente el opuesto. Por esta razón, el estudio del contexto es necesario, aunque no sea tarea fácil.

- **Comparaciones.** Las comparaciones son otro desafío en el análisis de sentimientos y se explicará haciendo uso de los siguientes ejemplos: "*Este helado es insuperable*", "*Este helado está mejor que los anteriores", "Mejor esto que nada"*. La primera comparación no necesitaría de un contexto para obtener su sentimiento correctamente, mientras que en las otras dos la situación puede variar. En el segundo caso, si se consideran que los helados anteriores eran pésimos, se podría estar hablando de un sentimiento neutro, al igual que el tercero. Por tanto, se puede apreciar que existe una relación directa con el contexto, descrito en el apartado anterior.

Capítulo 3: Análisis del sistema

En este capítulo se describe la etapa de análisis del sistema a desarrollar en este trabajo, y se establecen las funcionalidades del mismo. Este sistema, cuyo desarrollo ha sido la parte principal de este trabajo, se ha denominado *MRT*. Así, en primer lugar, se describirán los requisitos de MRT y a continuación, se especificarán sus casos de uso, tanto en forma tabular como gráfica.

3.1 Establecimiento de los requisitos

En este apartado se establecen los requisitos del proyecto, que pueden ser tanto funcionales como no funcionales. Antes de la especificación se realizará una breve descripción para entender el formato en el que se van a exponer.

Identificador: RZ-XX			
Prioridad	Baja	Media	Alta
Nombre			
Versión			
Dependencias			
Descripción			
Necesidad	Opcional	Deseable	Esencial
Estabilidad	Baja	Media	Alta

Tabla 1: Plantilla para la especificación de requisitos

3.1.1 Descripción de los requisitos

En la *Tabla 1* se muestra el formato que sigue cada uno de los requisitos obtenidos para el trabajo

A continuación, se explica qué significa cada uno de los campos pertenecientes a la tabla modelo para la especificación de requisitos.

- **Identificador**: Código único que identifica al requisito y que facilita la trazabilidad. Los caracteres representan lo siguiente:
  - **R**: Requisito.
  - **Z**: Puede tomar los valores F y NF para indicar si se está trabajando con un requisito funcional o no funcional.
  - **X**: Toma valores numéricos que son únicos en cada uno de los dos tipos de requisitos.

- **Prioridad**: Conjunto de tres valores que indican la prioridad del cumplimiento del requisito en tres niveles: baja, media y alta. La prioridad nos sirve para indicar qué requisitos deberían cumplirse antes. Esto es crucial para la planificación global del proyecto.

- **Nombre**: Nombre descriptivo para que sea posible la identificación del requisito.

- **Versión**: Número de veces que el requisito se ha modificado durante la ejecución del proyecto

- **Dependencias**: Requisitos que intervienen durante la ejecución del requisito definido. Se indica mediante los identificadores de requisito correspondientes.

- **Descripción**: Descripción detallada pero no demasiado extensa de la funcionalidad que aporta el requisito.

- **Necesidad**: Indica la importancia del requisito para el desarrollo del proyecto en tres niveles: *Opcional, deseable y esencial*. **Opcional** correspondería con un requisito que puede no ser cumplido y no afectaría en gran medida al proyecto. **Deseable** sería un requisito cuyo cumplimiento sería bueno para el proyecto, pero no se garantiza. Si la necesidad fuese **esencial**, se hablaría de un requisito cuyo cumplimiento es obligatorio y que, sin él, el proyecto no se podría llevar a cabo.

- **Estabilidad**: Campo que cuenta con tres valores que indican la probabilidad de un futuro cambio en el requisito durante el desarrollo del proyecto: *baja*, *media* y *alta*.

Para la selección del valor en los campos donde sea necesario hacerlo, se deberá colorear y subrayar (ya que podría tratarse de una impresión en blanco y negro).

3.1.2 Requisitos funcionales

Los requisitos funcionales de un proyecto software especifican qué tiene que hacer dicho software. Cada requisito describe una funcionalidad que el sistema debe llevar a cabo, junto con el resto de información que le corresponda y que ha sido definida previamente.

Los requisitos funcionales del presente trabajo se describen a continuación, desde la *Tabla 2* hasta la *Tabla 24.*

Identificador: RF-01			
Prioridad	Baja	Media	Alta
Nombre	Conexión con la API de búsqueda de Twitter		
Versión	1.0		
Dependencias			
Descripción	El sistema debe ser capaz de conectarse a la API de búsqueda de Twitter para la sustracción de Tweets		
Necesidad	Opcional	Deseable	Esencial
Estabilidad	Baja	Media	Alta

Tabla 2: Requisito funcional RF-01

Identificador: RF-02			
Prioridad	Baja	Media	Alta
Nombre	Conexión con la API de streaming de Twitter		
Versión	1.0		
Dependencias			
Descripción	El sistema debe ser capaz de conectarse a la API de búsqueda de Twitter para la retransmisión de Tweets		
Necesidad	Opcional	Deseable	Esencial
Estabilidad	Baja	Media	Alta

Tabla 3: Requisito funcional RF-02

Identificador: RF-03			
Prioridad	Baja	Media	Alta
Nombre	Conexión con MongoDB		
Versión	1.0		
Dependencias			
Descripción	El sistema debe ser capaz de conectarse a MongoDB para el almacenamiento de Tweets		
Necesidad	Opcional	Deseable	Esencial
Estabilidad	Baja	Media	Alta

Tabla 4: Requisito funcional RF-03

Identificador: RF-04			
Prioridad	Baja	Media	**Alta**
Nombre	Integración de VaderSentiment		
Versión	1.0		
Dependencias			
Descripción	El sistema debe ser capaz de hacer uso de VaderSentiment para la obtención del sentimiento en el Tweet		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 5: Requisito funcional RF-04

Identificador: RF-05			
Prioridad	Baja	Media	**Alta**
Nombre	Creación de documento JSON		
Versión	1.0		
Dependencias	RF-01, RF-04		
Descripción	El sistema debe ser capaz de generar un documento formato JSON para que el cliente pueda hacer uso de la información obtenida		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 6: Requisito funcional RF-05

Identificador: RF-06			
Prioridad	Baja	Media	**Alta**
Nombre	Análisis en tiempo real		
Versión	1.0		
Dependencias	RF-02, RF-04		
Descripción	El sistema debe analizar en tiempo real los tweets sobre el tema que busque el usuario		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 7: Requisito funcional RF-06

Identificador: RF-07			
Prioridad	Baja	Media	**Alta**
Nombre	Búsqueda de Tweets		
Versión	1.0		
Dependencias	RF-01		
Descripción	El sistema debe ser capaz de encontrar Tweets ya publicados relacionados con la búsqueda ejecutada por el usuario		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 8: Requisito funcional RF-07

Identificador: RF-08			
Prioridad	Baja	Media	**Alta**
Nombre	Aplicación Web		
Versión	1.0		
Dependencias			
Descripción	El sistema debe ser capaz de interactuar con el usuario a través de una aplicación web		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 9: Requisito funcional RF-08

Identificador: RF-09			
Prioridad	Baja	Media	**Alta**
Nombre	Input en el sistema		
Versión	1.0		
Dependencias	RF-08		
Descripción	El usuario introducirá sus peticiones a través de la aplicación web por los campos que se establezcan		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 10: Requisito funcional RF-09

Identificador: RF-10			
Prioridad	Baja	Media	**Alta**
Nombre	Input inexistente		
Versión	1.0		
Dependencias	RF-08, RF-09		
Descripción	Si el usuario intenta ejecutar una búsqueda sin completar todos los campos, el sistema mostrará un mensaje solicitando que los rellene.		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 11: Requisito funcional RF-10

Identificador: RF-11			
Prioridad	Baja	Media	**Alta**
Nombre	Idioma de las búsquedas		
Versión	1.0		
Dependencias	RF-01, RF-02, RF-06, RF-07		
Descripción	El sistema debe ser capaz de recopilar Tweets en inglés.		
Necesidad	Opcional	**Deseable**	Esencial
Estabilidad	Baja	**Media**	Alta

Tabla 12: Requisito funcional RF-11

Identificador: RF-12			
Prioridad	Baja	Media	**Alta**
Nombre	Número de búsquedas		
Versión	1.0		
Dependencias	RF-01, RF-07		
Descripción	El sistema debe ser capaz de hacer tantas búsquedas sobre un tema como el usuario desee		
Necesidad	Opcional	**Deseable**	Esencial
Estabilidad	Baja	**Media**	Alta

Tabla 13: Requisito funcional RF-12

Identificador: RF-13			
Prioridad	Baja	Media	**Alta**
Nombre	Búsqueda y análisis de un tema		
Versión	1.0		
Dependencias	RF-01, RF-02, RF-06, RF-07		
Descripción	El sistema debe ser capaz de realizar una búsqueda y un análisis en tiempo real sin necesidad de reiniciar sobre un mismo tema		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 14: Requisito funcional RF-13

Identificador: RF-14			
Prioridad	Baja	Media	**Alta**
Nombre	Búsqueda y análisis de temas diferentes		
Versión	1.0		
Dependencias	RF-01, RF-02, RF-06, RF-07		
Descripción	El sistema debe ser capaz de realizar una búsqueda y un análisis en tiempo real sin necesidad de reiniciar sobre diferentes temas		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 15: Requisito funcional RF-14

Identificador: RF-15			
Prioridad	Baja	Media	**Alta**
Nombre	Botón de descarga		
Versión	1.0		
Dependencias	RF-05, RF-08		
Descripción	En el apartado de búsquedas, el sistema debe posibilitar la descarga del documento JSON generado durante la ejecución de la búsqueda		
Necesidad	Opcional	**Deseable**	Esencial
Estabilidad	Baja	**Media**	Alta

Tabla 16: Requisito funcional RF-15

Identificador: RF-16			
Prioridad	Baja	Media	**Alta**
Nombre	Información de la búsqueda		
Versión	1.0		
Dependencias	RF-01, RF-07, RF-08		
Descripción	El sistema debe mostrar información sobre la búsqueda relacionada con el número de tweets, retweets, fecha y hora del más reciente y del más antiguo y sentimiento medio del total		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 17: Requisito funcional RF-16

Identificador: RF-17			
Prioridad	Baja	Media	**Alta**
Nombre	Gráfica de procedencia de Tweets		
Versión	1.0		
Dependencias	RF-01, RF-02, RF-06, RF-07, RF-08		
Descripción	El sistema debe mostrar al finalizar una búsqueda o un análisis una gráfica que muestre la procedencia de los tweets recopilados		
Necesidad	Opcional	**Deseable**	Esencial
Estabilidad	Baja	**Media**	Alta

Tabla 18: Requisito funcional RF-17

Identificador: RF-18			
Prioridad	Baja	Media	**Alta**
Nombre	Gráfica en tiempo real		
Versión	1.0		
Dependencias	RF-02, RF-06, RF-08		
Descripción	En el análisis en tiempo real, la gráfica debe ser capaz de actualizarse sin que sea preciso actualizar la página para una cómoda visualización		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 19: Requisito funcional RF-18

Identificador: RF-19			
Prioridad	Baja	Media	**Alta**
Nombre	Tweet más positivo y negativo		
Versión	1.0		
Dependencias	RF-01, RF-07, RF-08		
Descripción	En el apartado de búsqueda, se mostrará el Tweet más positivo y negativo con su respectivo sentimiento		
Necesidad	Opcional	**Deseable**	Esencial
Estabilidad	Baja	**Media**	Alta

Tabla 20: Requisito funcional RF-19

Identificador: RF-20			
Prioridad	Baja	Media	**Alta**
Nombre	Gráfica de “Tweets por sentimiento”		
Versión	1.0		
Dependencias	RF-01, RF-02, RF- 06, RF-07, RF-08		
Descripción	El sistema debe mostrar al finalizar una búsqueda o un análisis una gráfica que muestre el número de Tweets pertenecientes a cada sentimiento, pudiendo ser este Negativo, Neutral o Positivo		
Necesidad	Opcional	**Deseable**	Esencial
Estabilidad	Baja	**Media**	Alta

Tabla 21: Requisito funcional RF-20

Identificador: RF-21			
Prioridad	Baja	Media	**Alta**
Nombre	Reinicio de gráficas		
Versión	1.0		
Dependencias	RF-01, RF-07, RF-08		
Descripción	Las gráficas deberán reiniciarse al ejecutarse una nueva búsqueda		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 22: Requisito funcional RF-21

Identificador: RF-22			
Prioridad	Baja	Media	**Alta**
Nombre	Almacenamiento de Tweets		
Versión	1.0		
Dependencias	RF-01, RF-02, RF-03, RF-06, RF-07		
Descripción	Todos los tweets deberán quedar almacenados en mongoDB, siendo cada colección un tema que se haya buscado		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 23: Requisito funcional RF-22

Identificador: RF-23			
Prioridad	Baja	Media	**Alta**
Nombre	Actualización de las gráficas		
Versión	1.0		
Dependencias	RF-01, RF-07, RF-08		
Descripción	Las gráficas deben ser capaz de actualizarse con cada búsqueda sobre un mismo tema		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 24: Requisito funcional RF-23

3.1.3 Requisitos no funcionales

Los requisitos no funcionales describen características del funcionamiento del software, es decir, cómo se cumplen esas funcionalidades necesarias para el correcto funcionamiento del proyecto.

Los requisitos funcionales del presente trabajo se describen a continuación, de la *Tabla 25* hasta la *Tabla 29*.

Identificador: RNF-01			
Prioridad	Baja	Media	**Alta**
Nombre	Lenguaje de programación		
Versión	1.0		
Dependencias			
Descripción	El programa deberá ser desarrollado en Python debido de su compatibilidad con las librerías utilizadas		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 25: Requisito no funcional RNF-01

Identificador: RNF-02			
Prioridad	Baja	Media	**Alta**
Nombre	Compatibilidad con navegador		
Versión	1.0		
Dependencias			
Descripción	El programa deberá ejecutarse en Chrome		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 26: Requisito no funcional RNF-02

Identificador: RNF-03			
Prioridad	Baja	Media	**Alta**
Nombre	Requests de la API de Twitter		
Versión	1.0		
Dependencias			
Descripción	Se podrán ejecutar hasta 450 requests cada 15 minutos de la Search API de Twitter		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 27: Requisito no funcional RNF-03

Identificador: RNF-04			
Prioridad	Baja	Media	**Alta**
Nombre	Antigüedad de los Tweets		
Versión	1.0		
Dependencias			
Descripción	Los Tweets provenientes de las búsquedas podrán ser de hasta 1 semana de antigüedad		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 28: Requisito no funcional RNF-04

Identificador: RNF-05			
Prioridad	Baja	Media	**Alta**
Nombre	Formato de archivos de Twitter API		
Versión	1.0		
Dependencias			
Descripción	El formato de los archivos que provienen de Twitter con los que trabajará la aplicación, será JSON		
Necesidad	Opcional	Deseable	**Esencial**
Estabilidad	Baja	**Media**	Alta

Tabla 29: Requisito no funcional RNF-05

3.2 Casos de uso

A continuación, se procede a definir los casos de uso. Un caso de uso es una descripción de los pasos a seguir por parte de un actor, en este caso, uno de los usuarios que hacen uso del sistema, para la realización de una tarea concreta en el sistema.

3.2.1 Descripción tabular de los casos de uso

El formato tabular será el siguiente:

Identificador: CU-XX	
Título	
Actores	
Objetivo	
Precondiciones	
Postcondiciones	
Condiciones de error	

Tabla 30: Plantilla para la especificación de casos de uso

A continuación, se explica el significado de cada uno de los atributos que componen la tabla:

- **Identificador**. Código único de cada caso de uso que sigue el formato CU-XX donde:
  - CU: Indica que es un caso de uso.
  - XX: Representa el número del caso de uso.

- **Titulo**. Nombre del caso de uso.
- **Actores**. Sujetos que interactúan con el sistema en el caso de uso.
- **Objetivo**. Acción que se quiere lograr por parte de los actores.
- **Precondiciones**. Condiciones que se tienen que cumplir para que se pueda iniciar el caso de uso.
- **Postcondiciones**. Estado en el se encuentra el sistema tras acometer la acción y por tanto el caso de uso
- **Condiciones de error**. Situaciones que al tratar de llegar al objetivo pueden producir una excepción en el sistema

Los casos de uso son los que se muestran a continuación.

Identificador: CU-01	
Título	Búsqueda de datos
Actores	Usuario
Objetivo	Realizar satisfactoriamente la búsqueda de datos respecto a un tema concreto.
Precondiciones	1) Pulsar el botón de "Search now" 2) Rellenar los campos que se muestran
Postcondiciones	Página con los resultados obtenidos.
Condiciones de error	No se rellena alguno de los campos que se exigen.

Tabla 31: Caso de uso CU-01

Identificador: CU-02	
Título	Análisis de datos en tiempo real
Actores	Usuario
Objetivo	Realizar satisfactoriamente un análisis en tiempo real sobre un tema concreto.
Precondiciones	1) Pulsar el botón de "Analyze now" 2) Rellenar los campos que se muestran
Postcondiciones	Página con los resultados obtenidos,
Condiciones de error	No se rellena alguno de los campos que se exigen

Tabla 32: Caso de uso CU-02

Identificador: CU-03	
Título	Repetir una búsqueda de datos sobre un mismo tema
Actores	Usuario
Objetivo	Repetir satisfactoriamente una búsqueda de datos sobre un tema buscado previamente
Precondiciones	1) Haber realizado una búsqueda del mismo tema anteriormente 2) Pulsar en el botón que hace referencia al objetivo 3) Introducir número de tweets
Postcondiciones	Página con los resultados obtenidos, que se añaden a los ya existentes
Condiciones de error	Se repite una búsqueda desde la página de inicio

Tabla 33: Caso de uso CU-03

Identificador: CU-04	
Título	Descargar el fichero JSON que contiene los datos
Actores	Usuario
Objetivo	Descargar el fichero JSON que contiene los datos de la búsqueda que se ha realizado
Precondiciones	1) Haber realizado una búsqueda 2) Pulsar en el botón que hace referencia al objetivo
Postcondiciones	Descarga del fichero, que se podrá ver en las descargas del navegador
Condiciones de error	

Tabla 34: Caso de uso CU-04

Identificador: CU-05	
Título	Arranque del sistema
Actores	Desarrollador
Objetivo	Dejar el sistema preparado para su correcto funcionamiento cuando el usuario haga uso de él
Precondiciones	1) Inicializar base de datos 2) Ejecutar código
Postcondiciones	Funcionamiento del sistema en Localhost
Condiciones de error	Si la base de datos no se inicializa, al realizar una búsqueda o análisis se producirá un error, ya que todos los datos quedan almacenados en servidor.

Tabla 35: Caso de uso CU-05

3.2.2. Descripción gráfica de los casos de uso

En este apartado se presentan los casos de uso de forma gráfica. El objetivo es permitir la observación de las relaciones entre los casos de uso y los actores.

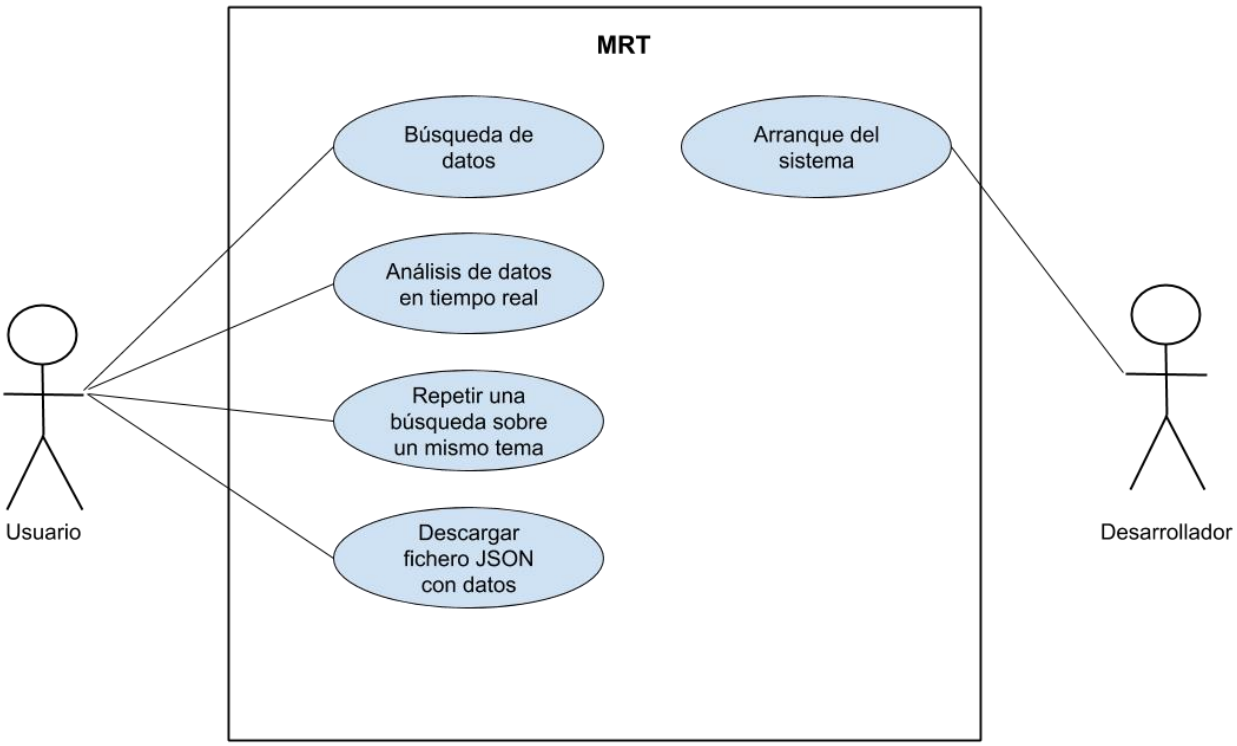

Ilustración 6: Casos de uso

Capítulo 4: Arquitectura y diseño del sistema

En este capítulo se describe el proceso de diseño de MRT, el sistema desarrollado. Este proceso tiene como objetivo la definición de la arquitectura del sistema, dividiéndolo en componentes y especificando detalladamente cada uno de ellos.

Para ello, se irá desde un nivel mayor de abstracción, hablando del sistema como conjunto, a un nivel más detallado, explicado cada uno de los módulos que divide la arquitectura de la plataforma.

4.1 Arquitectura general del sistema

A continuación, en la *Ilustración 7* se muestra el diagrama con los diferentes componentes que representa la arquitectura del sistema al completo, junto con números que indican el orden de las interacciones que se efectúan entre ellos.

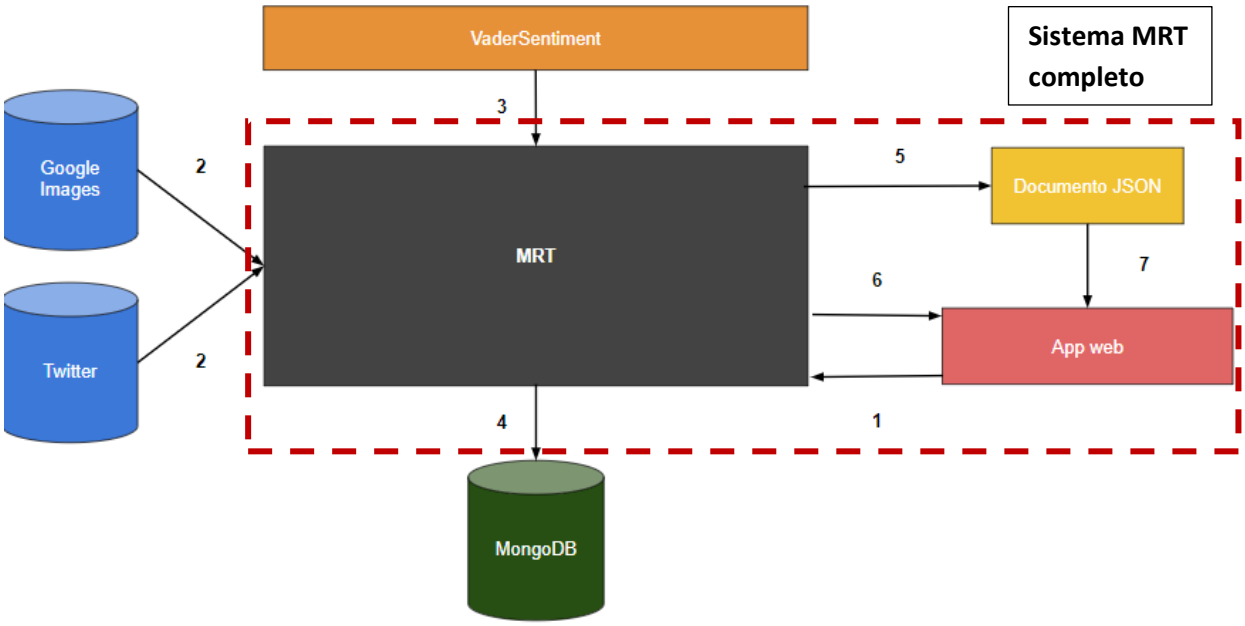

Ilustración 7: Arquitectura general

Como se aprecia en la imagen, existen múltiples módulos que se comunican con MRT para el correcto funcionamiento del sistema. Dichos módulos se describen a continuación:

- **Twitter API**. Gracias a la API que Twitter ofrece, se es capaz de extraer los datos con los que se trabajará posteriormente. Para acceder a estos datos que se proporcionan, se hace uso de Twython [27], una librería con la que se accede rápida y cómodamente, y que es utilizada por empresas e instituciones educativas. Twython será la vía de comunicación con Twitter, y todas las búsquedas y análisis en tiempo real pasarán por ahí. Los Tweets que se reciban

estarán en formato JSON y será entonces cuando se decidirá con qué campos contar para el análisis y cuáles se desecharan.

- **Google images**. Siguiendo la *Ilustración 7* de izquierda a derecha, se pasa a hablar de este módulo. Google Images es utilizada para una pequeña funcionalidad que se recoge en el apartado de búsquedas. Gracias a una librería incorporada al sistema que funciona a modo de *Scraper*, se recoge de forma automática imágenes. De esta forma, cuando el usuario realice una búsqueda, encontrará una imagen de la temática en la página de búsquedas, dándole una mayor consistencia.

- **VaderSentiment.** Se trata de una herramienta de análisis de sentimientos de código abierto [28]. Con esta librería se es capaz de identificar el sentimiento de cualquier texto, dando un valor de entre -1 y 1, siendo el primero un comentario muy negativo y el segundo muy positivo. En el caso de que el valor fuera 0, se estaría hablando de un comentario neutro. Cabe destacar que también puede traducir emojis y expresiones que se encuentran con frecuencia en redes sociales, por lo que es de gran utilidad para este proyecto.

- **MRT.** El módulo que conecta con todos los demás y con el que se tiene la capacidad de desplegar la aplicación web para que el usuario pueda hacer uso de la plataforma. Desde aquí se extraen los datos que el cliente requiere de Twitter, se analizan, se guardan en la base de datos y se despliegan para ser visualizados en diferentes gráficas. También es aquí desde donde se genera el fichero JSON que podrá descargarse.

- **MongoDB.** Base de datos NOSQL orientado a documentos donde se almacenarán todos los datos consultados [29]. Desde el lado del cliente no se tiene acceso, pero desde el del servidor servirá para consultar cualquier cosa que se pudiese considerar relevante. Cada uno de los documentos corresponderá con un Tweet con los siguientes campos:
    - **_id:** Identificador del documento, que es único y no se puede repetir dentro de la misma colección (conjunto de documentos).
    - **Type:** Campo en el que se especifica si el documento es un Tweet o un Retweet.
    - **Text:** Texto que se comparte en la red social por parte de un usuario. Input para VaderSentiment.
    - **Time**: Momento en el que se compartió el Tweet. En el módulo MRT se implementa un método para adaptarlo al huso horario del cliente, en este caso CEST.

- **Sentiment**: Sentimiento del texto del Tweet que ha sido analizado. Como se explica anteriormente, se trata de un valor numérico con varios decimales que comprende el rango de -1 y 1.

En la *Ilustración 7* se enumera el orden en el que se van a producir las interacciones entre los distintos módulos cuando se ejecute una búsqueda. Se procede a describir cada una de ellas:

- **1.** El usuario, a través de la aplicación Web, elegirá el tipo de funcionalidad de la que quiere hacer uso, pudiendo ser las dos comentadas. A continuación, procederá a introducir el tema que quiere buscar, el número de tweets y el idioma de estos.
- **2.** Con los datos obtenidos a través de la App, se extraen los datos de Twitter a través de Twython y se recoge de cada tweet que se recibe en formato JSON solo lo imprescindible, que corresponde con los campos mencionados en el módulo MongoDB.
- **3.** A continuación se procede al análisis del texto de cada tweet recibido, creando además un nuevo campo que corresponde con el sentimiento. También se ejecutan otras muchas operaciones que se describirán con más detalle en el *capítulo 5: Implementación del sistema*.
- **4.** Seguidamente se almacenan los Tweets en la base de datos por si fueran de interés en un futuro.
- **5.** Además de almacenar en MongoDB, se escribe en un documento JSON lo mismo que se está almacenando en la base de datos para que el usuario lo pueda descargar (**Interacción 7**)
- **6.** Por último se vuelcan los datos a la plataforma Web para que el usuario lo pueda visualizar. Será necesario el uso de *websockets* para el envío entre servidor y cliente sin que sea necesario refrescar la página. De este tema también se entrará en más detalle con posteridad, en el *capítulo 5: Implementación del sistema*.
- **7.** Como se explica en la **Interacción 5,** ésta corresponde con la posibilidad de descarga del cliente del documento JSON.

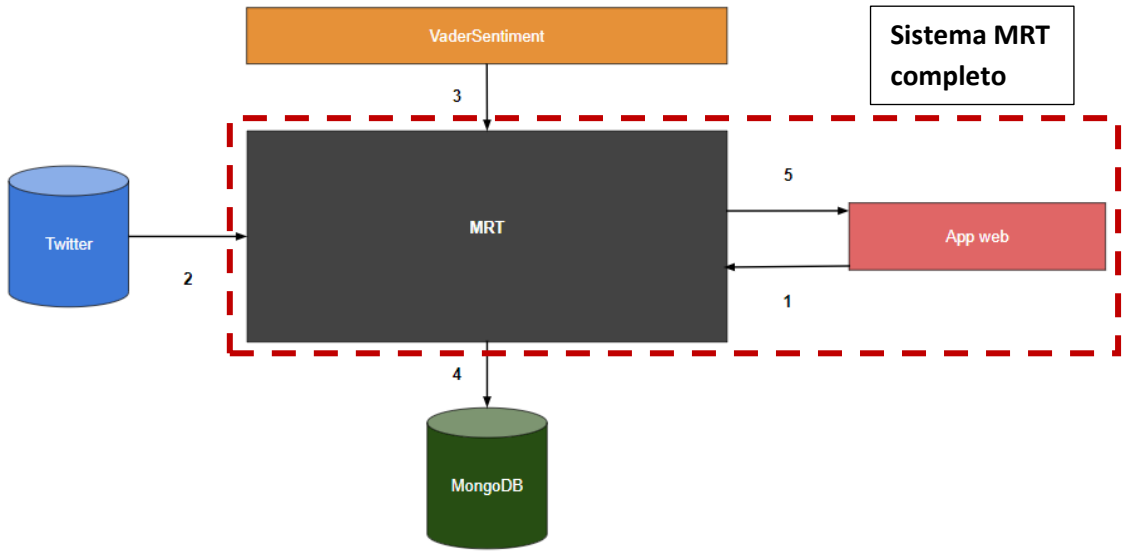

Ilustración 8: Sistema en el análisis en tiempo real

En la *Ilustración 8* se muestra cómo trabajaría el sistema en el caso de que se hiciese un análisis en tiempo real. Como se puede observar, en esta situación, el módulo de Google Images y el del documento JSON no serían utilizados, pues se ha decidido no contar con estas funcionalidades.

4.2 Arquitectura del sistema MRT

El sistema MRT cuenta con diferentes subsistemas que serán abordados en el próximo capítulo. Aquí, bastará con hacer una breve descripción del funcionamiento de cada uno para entender su funcionamiento. A continuación, se describen los diferentes componentes de los que consta MRT:

- **Comunicación con la App web**: En este componente se encuentran todos los mecanismos que permiten la recogida de datos a través de la aplicación y el posterior envío con los resultados. Es el canal de comunicación entre los dos lados, a excepción una pequeña característica con la que cuenta el componente de análisis en tiempo real, que se explicará posteriormente.

- **Búsqueda**: Componente encargado de recoger la información de la API, analizarla, almacenarla y enviar de vuelta. Entra en contacto con todos los módulos del sistema a excepción de la aplicación Web. Durante el siguiente capítulo se explicará con más detenimiento las diferentes tareas que lleva a cabo.

- **Análisis en tiempo real:** Componente del sistema MRT que extrae los Tweets en el mismo momento que son publicados. Al igual que en el componente anterior, se encarga de analizar, almacenar en la base de datos y devolver el resultado para la visualización del usuario.

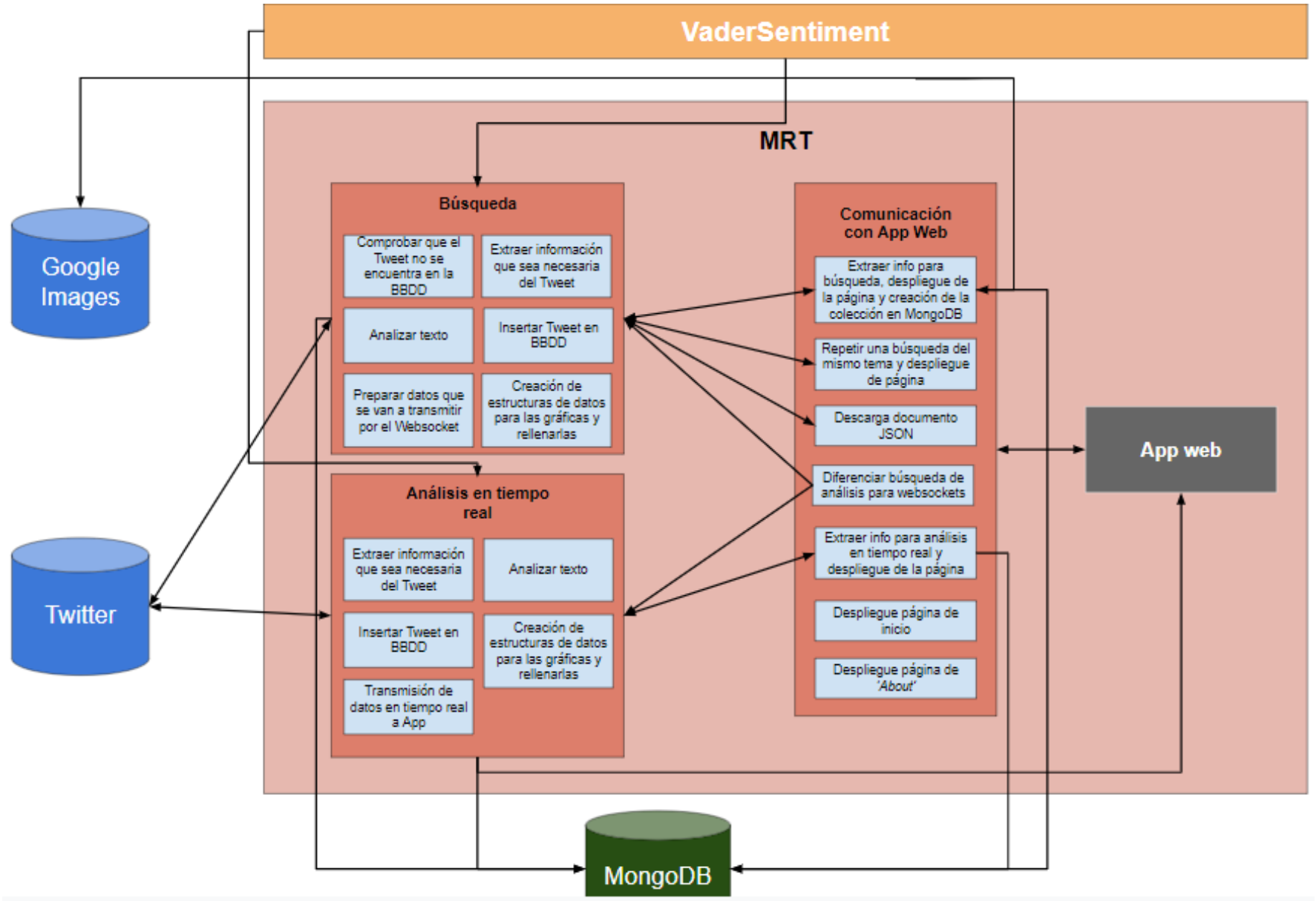

Ilustración 9: Arquitectura del sistema MRT

Capítulo 5: Implementación del sistema

En este capítulo se describe de forma detallada el proceso seguido para la implementación de cada uno de los bloques que componen el sistema, MRT.

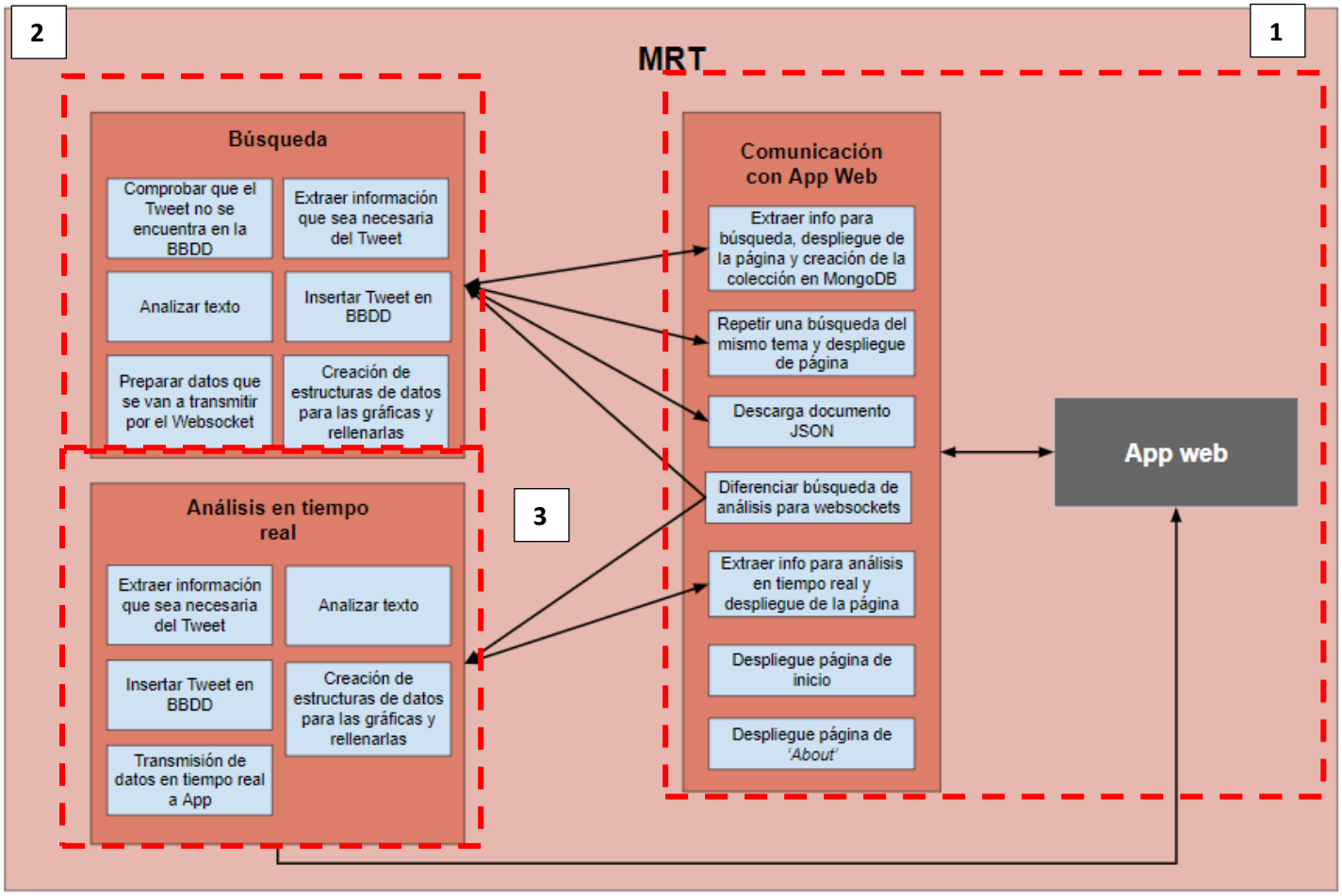

Ilustración 10: MRT

5.1 Bloque 1: App Web y comunicación con el resto del sistema

En este bloque se detalla con más profundidad cómo nuestro sistema es capaz de lanzar una aplicación Web y cómo se comunica usuario y servidor haciendo uso de *websockets*.

5.1.1 Aplicación web

Para la aplicación Web se hace uso de Flask, un micro-framework basado en lenguaje Python que permite desarrollar aplicaciones de forma sencilla con un mínimo número de líneas de código [30]. Se le considera micro porque, aunque bien es cierto que es muy fácil desarrollar una App, no cuenta con demasiadas extensiones, lo cual puede limitar en ocasiones. Es por ello, por lo que se hace uso de una extensión, Flask-Bootstrap [31], para

el Frontend de la plataforma. Bootstrap es otro *Framework* muy utilizado, en este caso para el diseño Web [32]. La experiencia del usuario es algo que se ha tenido en cuenta en este proyecto y tras comprobar su compatibilidad con Flask, se ha decidido implementarlo, facilitando y mejorando notoriamente el diseño de la plataforma tras utilizarlo.

Otras herramientas que se han utilizado para mejorar la experiencia de los usuarios de cara al visionado de los datos analizados han sido Plotly y Chart.js. Ambas ayudan a generar fácilmente gráficas responsivas e interactivas basadas en Javascript. En el caso de Plotly [33], las gráficas creadas cuentan con características como el auto-escalado, posibilidad de hacer o deshacer zoom, descargar la gráfica como PNG… en definitiva, todo lo que ayude al cliente a estudiar los resultados obtenidos. Chart.js ha servido para producir diversas gráficas sectoriales o "*pie charts*" con la que visualizar información relacionada con la procedencia de los tweets y cuántos de estos pertenecen a un sentimiento u otro.

5.1.2 Comunicación con el sistema

Otra de las ventajas de Flask es que incluye un servidor Web de desarrollo, por lo que no se necesita ninguna otra infraestructura para poder ejecutar el código y por tanto permite ver fácilmente lo que se obtiene mientras está en ejecución. Además, cuenta con un depurador que señala en caso de error dónde puede estar ocurriendo. Por tanto, es sencillo implementar todo el código del sistema en un mismo lugar.

En este bloque se explicará la conexión que existe entre la App y el mecanismo que hay detrás para que el usuario pueda visualizar lo que solicita. Como se puede ver en la *Ilustración 10*, existen múltiples tareas que tienen que ocurrir para el correcto funcionamiento, y que se irá explicando detalladamente sin llegar a hablar del código en sí. Se explica a continuación uno a uno siguiendo un orden de arriba-abajo:

- **Extracción de información de búsqueda, despliegue y creación de colección**: La primera característica es la capacidad de extraer de la aplicación web los datos con los que el resto del sistema va a trabajar. Para ello, el usuario deberá contar con un formulario dónde rellene los campos que se le pidan en función de lo que quiera hacer. Una vez hecho esto, se enviará una petición HTTP al servidor, que recibirá los parámetros y se procederá con los siguientes métodos. Cabe destacar, y es algo común para la página donde se lleva a cabo el análisis en tiempo real también, que primero se despliega la página de resultados vacía, exceptuando la imagen asociada a la búsqueda, pues ya está incorporada cuando se despliega. Es después, cuando cliente comunica a servidor que se trata de una búsqueda, cuando se vuelcan los datos en la ruta. Para ello se hace uso de *websockets*, algo que se explicará más adelante. Por último, se crea la colección en la base de datos, que llevará el nombre la temática buscada.

- **Nueva búsqueda sobre un mismo tema**: La siguiente funcionalidad es la capacidad de repetir búsquedas sobre un mismo tema cuantas veces quiera el usuario. Debido a que se utiliza la versión gratuita de la API de Twitter de búsqueda, no es posible extraer más de 100 Tweets por petición. Por tanto, la posibilidad de que quien haga uso de nuestra plataforma pueda recopilar más de 100 tweets era algo que debía añadirse. Para ello, se diferencia una petición de búsqueda de un nuevo tema de la de una igual, a través del botón que se incorpora en la página de resultados. Si el usuario busca más Tweets, accionará ese botón, por el cuál se será capaz de diferenciar. Hay que tener en cuenta que en una nueva búsqueda sobre un mismo tema puede ocurrir que se encuentren tweets ya recogidos. Como el código que identifica cada tweet es único, se comprueba que el que se va a añadir no se encuentra en la base de datos. No será necesario acceder a la BBDD, pues se cuenta con una lista que los va almacenando y que sólo se reinicia cuando se busca algo nuevo. Cuando una búsqueda se repite, no se borran los valores de las variables globales, ya que si no sólo se mostraría la información que corresponde a la nueva búsqueda. Por tanto, el usuario verá todos los datos recopilados desde la primera petición y si lo desea, contará con todos los Tweets al descargar el documento JSON.

- **Descarga del documento JSON**: Como ya se comenta en el apartado anterior, se da la posibilidad al usuario de descargar un documento JSON que recopile todos los Tweets de los temas buscados. Los tweets se vuelcan en el documento con cada búsqueda. Provienen de una estructura de tipo diccionario que se reinicia o no dependiendo del tipo de petición, y que nunca sobrescribe el archivo, ya que, si no, el cliente sólo vería en el archivo la información proveniente de la última búsqueda realizada.

- **Diferenciar análisis de búsqueda a través de los Websockets:** La comunicación cliente-servidor es en la aplicación, el punto más importante de todos. Era necesario que esta comunicación fuera lo más rápida y suave posible, sin saltos bruscos ni interrupciones. En el caso de MRT, el protocolo HTTP es muy limitado, pues en el análisis en tiempo real que se verá a continuación, se necesitaba que la gráfica de visualización de tweets se actualizase constantemente sin necesidad de que el cliente hiciese una petición. A través del protocolo HTTP, la aplicación no hubiera sido posible, pues se debería actualizar la página constantemente sin que el usuario pudiese ver lo que está sucediendo.

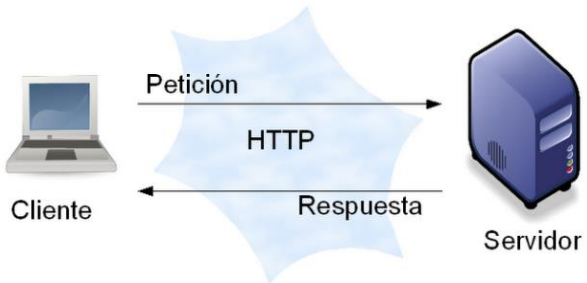

Ilustración 11: Comunicación Cliente-Servidor mediante peticiones HTTP. Fuente: http://redes.noralemilenio.es/protocolos-del-nivel-de-aplicacion/?upm_export=print

En esta situación no sólo se estaría hablando de una peor experiencia del cliente, si no que sería imposible haber podido desarrollar la funcionalidad, y por lo tanto no habría experiencia alguna. Es en este momento donde los *websockets* son utilizados para aportar esa tecnología necesaria para llevar el proyecto a cabo.

Los *websockets* permiten el flujo de información en ambos sentidos en el mismo instante en el que ocurren los eventos. No se limita a la petición de uno y respuesta del otro. Es una comunicación bidireccional entre ambos.

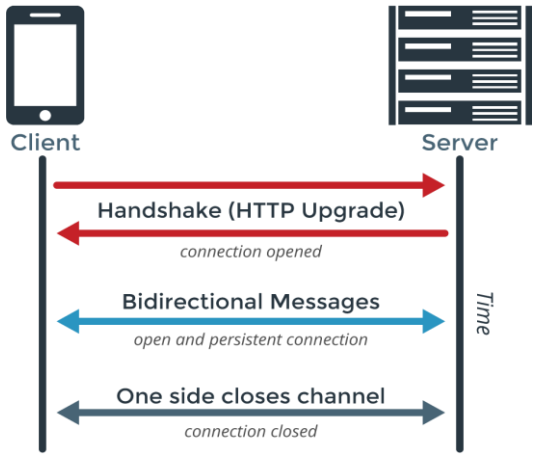

Ilustración 12: Protocolo de comunicación mediante websockets. Fuente: https://www.pubnub.com/blog/2015-01-05-websockets-vs-rest-api-understanding-the-difference/

Este aspecto es de vital importancia por las causas descritas anteriormente, pero también se le sacará partido para otras cuestiones. Por ejemplo, cuando se acceda a la página de resultados, ya sea la de análisis o búsqueda, se le enviará un mensaje a servidor preguntando cuál es. En búsquedas esto no es tan importante, ya que cuando se renderiza la página, los resultados ya se han obtenido. La diferencia sería, en este caso, los segundos que se espera para visualizar el contenido, que es el tiempo que tarda el websocket en transmitir la información. Pero, en el análisis no es lo mismo. La única forma de que la retrasmisión de tweets se ejecute nada más renderizar la página es a través de este mensaje. Al servidor le llega la palabra "live" y es a partir de ese momento en el que se ejecutará el método encargado de retransmitir los tweets.

La tecnología utilizada que emula a los *websockets* se llama SocketIO, y es una librería en Javascript. SocketIO también permite una comunicación bidireccional, basada en *websockets*.

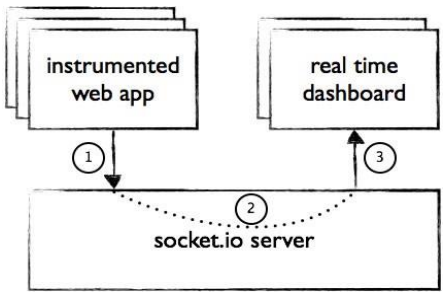

Ilustración 13: Integración de socket.io en aplicaciones web. Fuente: https://medium.com/@eddydecena/servidor-real-time-con-socket-io-18e84d39d12b

- **Extracción de información de análisis, despliegue y creación de colección:** Se obtienen los datos de la misma forma que con la búsqueda, a través del rellenado de campos y su envío tras accionar el botón. A continuación, se despliega la página de resultados de análisis vacía, ya que como se explica en el apartado anterior, una vez el cliente se encuentra en esta página se comunicará al servidor que es un análisis. Antes de este mensaje se habrá creado la colección en la base de datos cuyo nombre es el tema buscado. El análisis de los tweets se detallará en el bloque correspondiente.

- **Despliegue de la página de inicio:** Se trata del renderizado de la página principal nada más entrar en la dirección asignada, que en este caso será Localhost.

- **Despliegue de la página de "About":** Al igual que ocurre con las demás, si el usuario quisiera entrar aquí se procedería a renderizar la página.

5.2. Bloque 2: Búsqueda

En este bloque se explica con más profundidad el bloque referente al mecanismo de búsqueda.

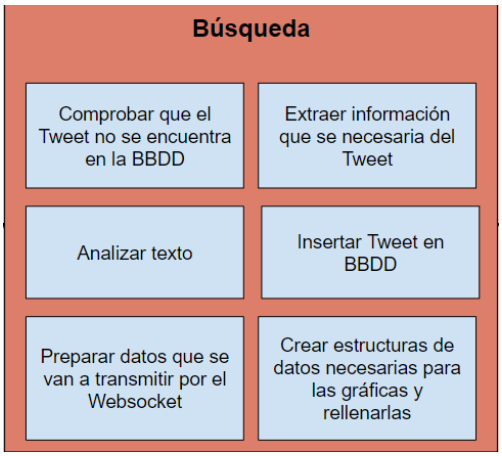

Ilustración 14: Bloque de búsqueda del sistema MRT

En este caso se describirá cada funcionalidad siguiendo un orden de izquierda-derecha:

- **Comprobar que el Tweet no se encuentra en la BBDD**: Esto será importante cuando se ejecuten varias búsquedas sobre un mismo tema, ya que, si no ha pasado demasiado tiempo y el tema no se comenta en exceso, pueden aparecer Tweets repetidos que podrían suponer un problema. En los análisis nunca podrá darse esta situación pues los Tweets están siendo publicados en el mismo momento que se está ejecutando el programa. Como se comenta en el bloque anterior, para comprobar que el Tweet no se encuentra en la BBDD, no se accede directamente a ella. Se cuenta con una lista que guarda los números de identificación de los

Tweets para verificar si se encuentra o no en ésta. La lista se reiniciará si se ejecuta una búsqueda de un tema nuevo.

- **Extraer información de utilidad del Tweet**: Antes de explicar cómo y cuáles son los campos que interesan para el análisis del tweet y que se van a extraer, se procede a explicar la estructura de los tweets y retweets [34]. También se explicará el cambio que supuso en noviembre de 2017 la aparición de los Tweets de 280 caracteres [35].

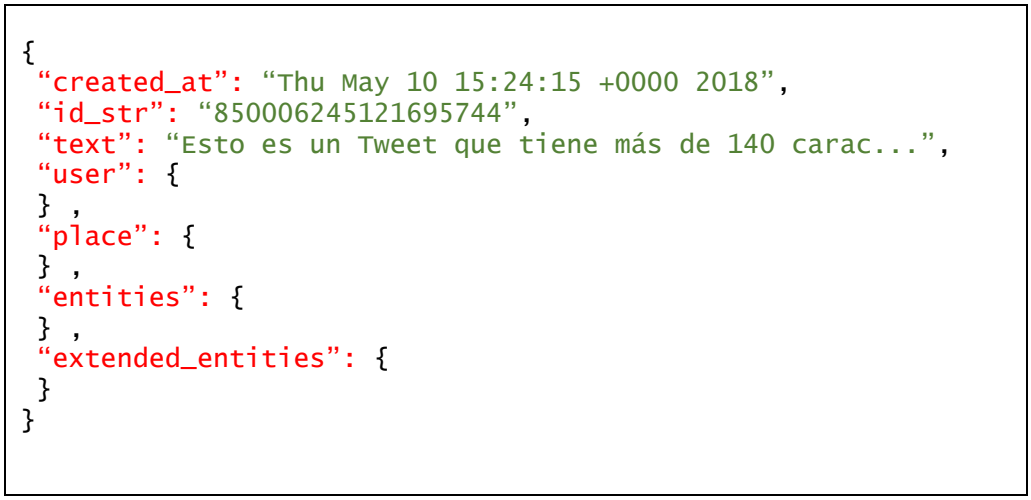

Ilustración 15: Estructura de un tweet

En la *Ilustración 15*, se observa la estructura de un tweet. Se trata de un dato JSON compuesto por diferentes objetos. Por ejemplo, todos los tweets incluyen el objeto 'User' que corresponde al autor del Tweet. Si el Tweet está etiquetado geográficamente, es decir, si el usuario permite la localización al publicar contenido, el campo 'Place' también aparecerá relleno. Cada Tweet además incluye un objeto de 'Entities' que encapsula campos destinados a Hashtags, menciones a otros usuarios, URLs o si se publica contenido multimedia, pudiendo ser imágenes, videos o GIFs. En este último caso, existiría un objeto 'Extended_entities'. Se cuenta también con objetos básicos como la fecha en la que fue creado, su número de identificación único y el texto.

En la siguiente imagen se observa cómo es la estructura de los Tweets que tienen más de 140 caracteres, es decir, de todos aquellos que se publicaron a partir de otoño de 2017. Como se puede apreciar, estos objetos cuentan con un nuevo campo 'Extended_tweet' para contener los mensajes más largos. Además, para referenciar que sobrepasan el anterior límite, 140, cuentan con un campo booleano 'truncated' que puede ser verdadero o falso si excede o no.

```
{
 “created_at”: “Thu May 10 15:24:15 +0000 2018”,
 “id_str”: “850006245121695744”,
 “text”: “Esto es un Tweet que tiene más de 140 carac…”,
 “display_text_range”: {0, 140}
 “truncated”: true,
 “user”: {
     “id_str”: “94456328”,
     “screen_name”: “unUsuarioCualquiera”
 } ,
 “extended_tweet”: {
     “full_text”: “Esto es un tweet que tiene más de 140
caracteres, #gracias, #Twitter”,
     “display_text_range”: [0, 248],
     “entities”: {
           “hashtags”: [{
                 “text”: “gracias”,
                 “índices”: [233, 240]
           },  {
                 “text”: “Twitter”,
                 “índices”: [241,248]
    },

 “entities”: {
    “hashtags”: []
  }

}
```

Ilustración 16: Estructura de los tweets con más de 140 caracteres

Se hará uso de esta estructura para aquellos tweets que sean originales, ya que engloba también a la anterior estructura.

Por último, se revisará la estructura de los retweets, que corresponde con la *Ilustración 17*. Estos, siempre contienen dos objetos. El Tweet original que se retuitea, y el tweet como tal, que es la repetición del original, pero incorporando datos de la persona que lo comparte. Retweetear es la acción de compartir y por tanto no se puede agregar ningún otro contenido. Es por eso por lo que el texto del Tweet será “RT @Autor texto original”. Además, no se puede conocer la ubicación de la persona que lo comparte, aunque del original sí. También es interesante destacar que, aunque se retuitee un retweet, el campo ‘Retweet_status’ siempre apuntará al original, por lo que si hay un intermediario no se conocerá. Con intermediario se hace alusión a la persona que compartió el contenido y por la cual otra persona compartió ese mismo contenido a través de ella.

```
{
 "Tweet": {
    "text": "RT @author Mensaje original",
    "user": {
         "screen_name": "retweeter"
    } ,
    "retweeted_status": {
       "text": "mensaje original",
          "user": {
                "screen_name": "TweeterOriginal"
          } ,
          "place": {
          } ,
          "entities": {
          } ,
          "extended_entities": {
          }
        } ,
       } ,
       "entities": {
       } ,
       "extended_entities": {
       }
      }
     }
```

Ilustración 17: Estructura de los retweets

Tras ver qué tipos de estructuras se encuentran al trabajar con la API de Twitter, se prodece a explicar cuáles han sido los campos que se han utilizado para el proyecto:

- **Id_str**: La representación en forma de String del identificador único del Tweet.
- **'User' -> 'Name':** Nombre de usuario de la persona que publica el contenido.
- **'Created_at':** Hora UTC del Tweet creado. Al ser el huso horario UTC es necesario transformarlo a CEST.
- **'Full_text':** Como se explica en la segunda estructura, muchos de los Tweets hoy en día sobrepasan el antiguo limite de 140 caracteres, por lo que se necesita el campo donde se encuentra el texto completo.

Realmente, para la finalidad del sistema propuesto, sólo haría falta el texto, ya que a partir de ahí se obtiene el sentimiento, pero es importante dotarlo de veracidad para que el usuario pudiese en cualquier momento encontrar el tweet si lo desease y ver que existe. Además, el identificador será utilizado a modo de clave primaria en la base de datos, ya que es único e irrepetible.

Para diferenciar si el tweet es original o compartido, sólo se tiene que hacer uso de un condicional. En el caso de que 'Retweet_status' fuese nulo, se estaría hablando de uno original, y si no, de un retweet. Del retweet se cogen los mismos campos, pero en este caso, para obtener el texto, se accede al objeto padre, que es el tweet original. Lo demás se obtiene del retweet, ya que si no se estaría repitiendo el identificador y se podría tener problemas.

- **Analizar texto (obtención del sentimiento)**: Para explicar cómo se analiza el texto se debe explicar cómo funciona VaderSentiment, que es la librería de código abierto utilizada, y de la que se obtiene un valor de entre -1 y 1. Este valor indica la negatividad o positividad del texto analizado. Este texto utilizado se limpia previamente, eliminando links, menciones a usuarios y caracteres especiales. Se procede pues a hablar del analizador de sentimientos.

  VADER se basa en un diccionario que asigna características léxicas a intensidades emocionales llamadas puntuaciones de sentimientos. La puntuación de sentimiento de un texto se puede obtener al resumir la intensidad de cada palabra en el texto. Por característica léxica, nos referimos a cualquier cosa utilizada para la comunicación textual. En un tweet típico, se pueden encontrar no solo palabras, sino también emoticonos como ":-)" o siglas como "LOL". Lo bueno del análisis de sentimientos de VADER es que estos coloquialismos también se asignan a los valores de intensidad.

  Se puede pensar que la intensidad emocional es muy arbitraria, ya que depende de a quién se le pregunte. Lo que para una persona puede no ser algo muy negativo, para otra sí que lo es. Para contrarrestar esto, los creadores del análisis de sentimiento VADER reclutaron no solo a uno, sino a varios evaluadores humanos de *Amazon Mechanical Turk* y promediaron sus calificaciones para cada palabra. Esto se basa en el concepto de la sabiduría de la multitud: la opinión colectiva a menudo es más confiable que la opinión individual.

  Las características léxicas no son las únicas cosas en la oración que afectan el sentimiento. Hay otros elementos contextuales, como la puntuación, el uso de mayúsculas y los modificadores, que también dan emoción. El análisis de la opinión de VADER los tiene en cuenta al considerar varias heurísticas simples. El efecto de estas heurísticas es cuantificado de nuevo utilizando evaluadores humanos. A continuación, se detallarán algunas de ellas:

  - **Puntuación**: Se compara 'Me gusta' y '¡Me gusta!'. La segunda tiene más intensidad que la primera, y por tanto la puntuación del sentimiento tiene que ser más alta. VADER tiene esto en cuenta. Para ello, primero calcula la puntuación de la oración sin prestar atención a las exclamaciones. Una vez ose obtiene un valor, resta o suma en función del resultado anterior.

- **Capitalización**: Se compara ahora ‘Una actuación INCREÍBLE’ de ‘una actuación increíble’. La segunda definitivamente tiene más intensidad, y VADER lo tiene en cuenta, sumando o restando en función del resultado final.

- **Cambio de polaridad debido a ‘pero’**: A menudo, ‘pero’ conecta dos cláusulas con sentimientos que se contrastan. El sentimiento, sin embargo, tiende siempre a ser el último. Por ejemplo, ‘Me gustó la película, pero no la quiero ver de nuevo’. La primera cláusula ‘Me gustó la película’ es positiva, pero la segunda ‘pero no quiero verla de nuevo’ es negativo. Es obvio que la segunda es la dominante. VADER reduce en un 50% el valor de la primera cláusula y aumenta en un 150% la segunda.

Estas serían algunas de las heurísticas que utiliza VADER. Por tanto, el valor obtenido es el resultado de combinar un diccionario de características léxicas con el conjunto de heurísticas. El modelo funciona por lo general mejor cuando se aplica en textos de redes sociales, como los que se utilizan en el proyecto, que grandes textos como artículos de opinión que se puedan encontrar en periódicos.

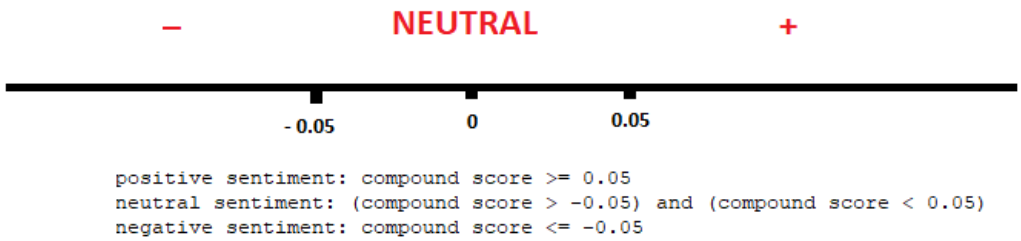

Ilustración 18: Sentimiento en función del valor obtenido.
Fuente: https://medium.com/@rahulvaish/vader-and-sentiment-analysis-python-eae70ecef454

- **Insertar Tweet en la base de datos**: Como se ha explicado anteriormente, MongoDB será la base de datos utilizada. Para ello, se utiliza Pymongo [36], una librería que nos conecta nuestra aplicación desarrollada en Python con la BBDD. El hecho de utilizar una base de datos no relacional es debido entre otras cosas a que no era necesaria una conexión entre colecciones. Cada una de ellas es un tema buscado por el usuario, y sus respectivos documentos, los Tweets obtenidos. También se buscaba una tecnología de almacenamiento de datos que fuera fácil de implementar y veloz. Los documentos son objetos JSON, que dentro de MongoDB se conocen como BSON, binary JSON. Por tanto, tras saber la forma con la que se va a almacenar la información, se extrae. A continuación, se analiza el texto, y se crea un nuevo campo llamado ‘Sentiment’ para almacenar el sentimiento obtenido. Tras esto, se guarda en la BBDD. Este proceso se repetirá tantas veces como Tweets se hayan pedido por parte del usuario.

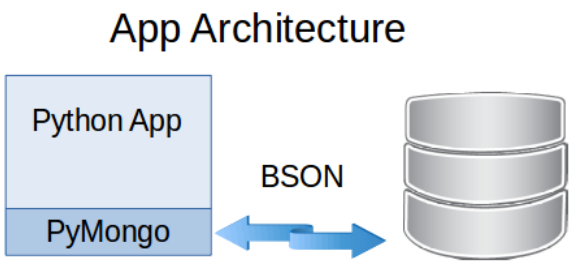

Ilustración 19: Comunicación entre aplicación y BBDD.
Fuente: https://www.bogotobogo.com

- **Preparar datos que se van a transmitir por el Websocket**: Este apartado complementa uno de los del bloque de comunicación con la App web, en el que se detallaba cómo el sistema tanto en el análisis en tiempo real como en la búsqueda esperaba al mensaje de cliente antes de transmitir. Cuando ya se ha realizado el análisis del tweet y guardado en la base de datos, se dejan todos los datos preparados para pasarlos por websocket cuando así se comunique. Estos datos son, como se ve en la *Ilustración 20*, el número de tweets totales, número de tweets originales del total, número de retweets del total, el sentimiento medio de todos ellos y la fecha del más antiguo y del más reciente. Cabe destacar que en ocasiones no se recogen el número de Tweets especificado por usuario, y así se puede observar en estos datos que se facilitan. Tras múltiples pruebas se ha visto que el problema no reside en nuestra aplicación y puede deberse a Twython, la librería con la que se recoge información, o con la propia API de búsqueda de Twitter.

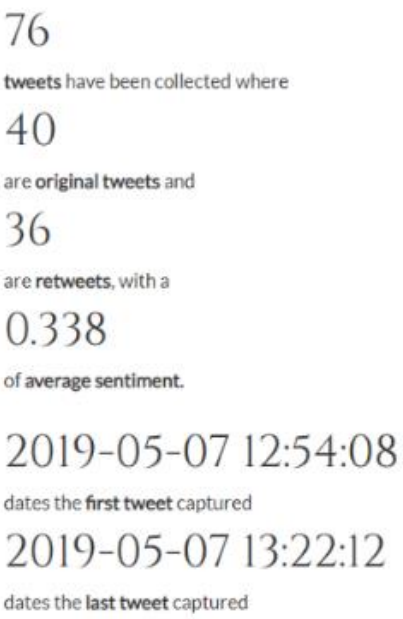

Ilustración 20: Información que se muestra de los datos obtenidos

- **Creación de estructuras necesarias para rellenar las gráficas**: Para finalizar este bloque correspondiente a la búsqueda, se explica este último apartado sobre

la creación de estructuras para el relleno de las gráficas que se utilizan. Al igual que se dejan en variables los valores preparados que se van a transmitir por los *websockets*, también ocurre lo mismo con las listas que van a utilizarse en las gráficas. Estas listas guardan los países de donde provienen los tweets y el número de positivos, negativos y neutros.

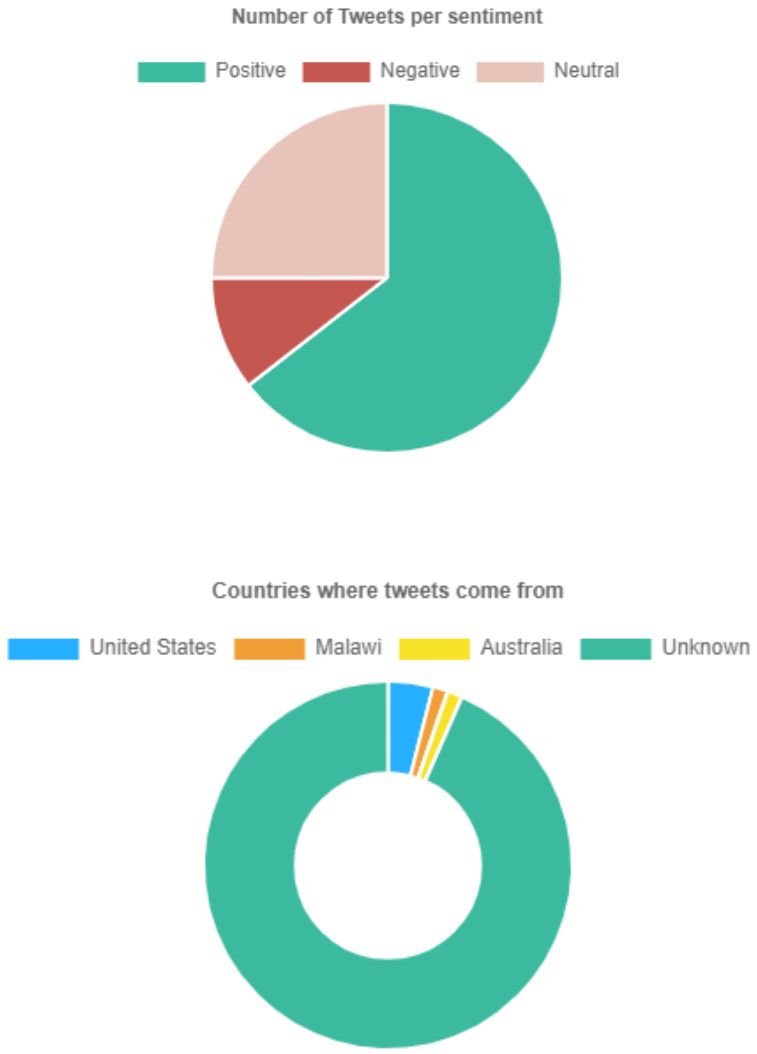

Ilustración 21: Gráficas que se muestran en la página de resultados

5.3. Bloque 3: Análisis en tiempo real

En este bloque se detalla con más profundidad el último bloque del sistema implementado, referente al mecanismo de análisis en tiempo real.

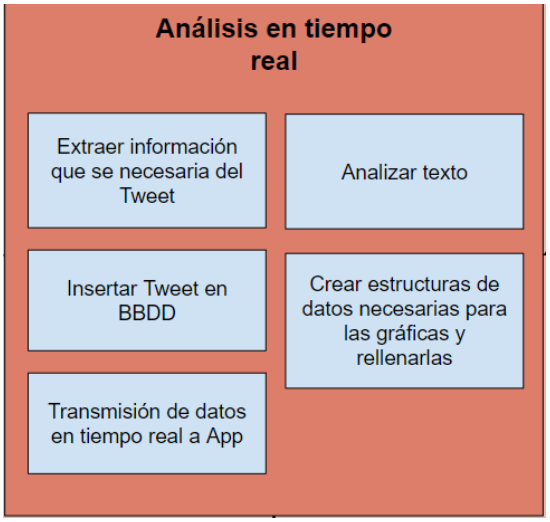

Ilustración 22: Bloque de ánalisis en tiempo real

En este caso, no se describirán todas las funcionalidades del bloque, pues todos a excepción de uno, han sido detallados en el bloque anterior. Aun así, cabe destacar que, en la extracción de información, solo se recogen los tweets publicados, no retweets. Por lo demás todo se mantiene de la misma forma. Se analiza el texto con VaderSentiment, se inserta en MongoDB a través de Pymongo, y se crean las estructuras necesarias para que al finalizar el análisis se desplieguen las gráficas estructurales.

No ocurre lo mismo con la gráfica principal, que se actualiza cada vez que en un nuevo Tweet entra. Es esta la principal causa del uso de *websockets* como se describe anteriormente. Recargar la página cada vez que se necesitaba actualizar la gráfica no era una opción válida, y con SocketIO y Plotly, aparece una solución viable. A continuación, se explica la última funcionalidad del bloque para entrar en más detalle:

- **Transmisión de datos en tiempo real a la aplicación Web**: A diferencia de la búsqueda, en la que los datos ya se habían obtenido antes de conectarse servidor y cliente, aquí no sucede de la misma manera. Una vez se recibe el mensaje de cliente especificando que se trata de un análisis en tiempo real, se ejecuta todo el proceso detallado en el bloque anterior: Extracción de información, obtención de sentimiento, inserción de Tweet y creación de estructuras. Alcanzado el número de Tweets que solicita el usuario, el método que conecta con la Api de retransmisión de Twitter se desconectará y se transmitirá a través del Websocket las estructuras para que se puedan generar las gráficas. Gracias a la velocidad del socket, esto se realizará en cuestión de décimas de segundo desde el momento que finaliza la retransmisión, sin que la experiencia de usuario pueda verse afectada.

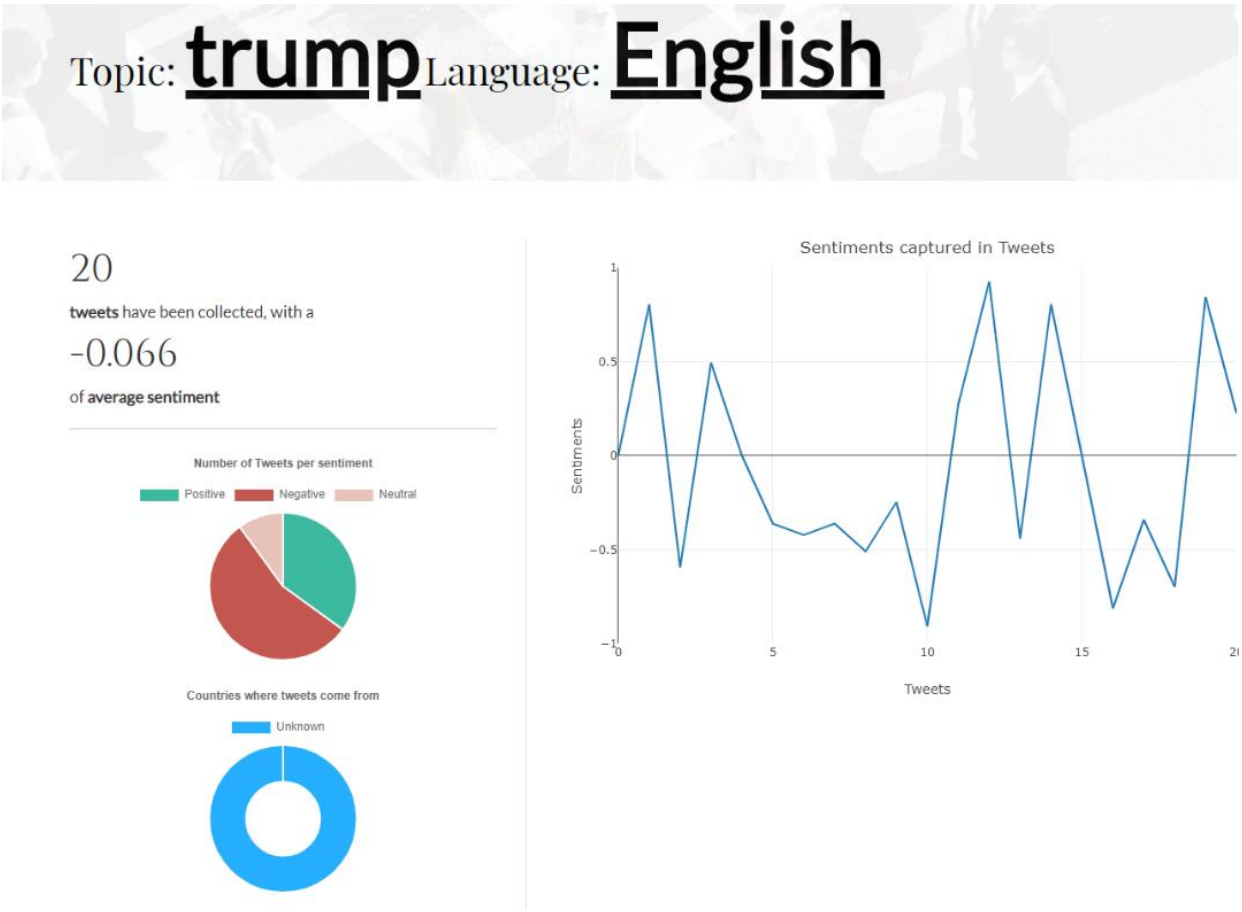

Ilustración 23: Página de resultados en tiempo real de la aplicación web

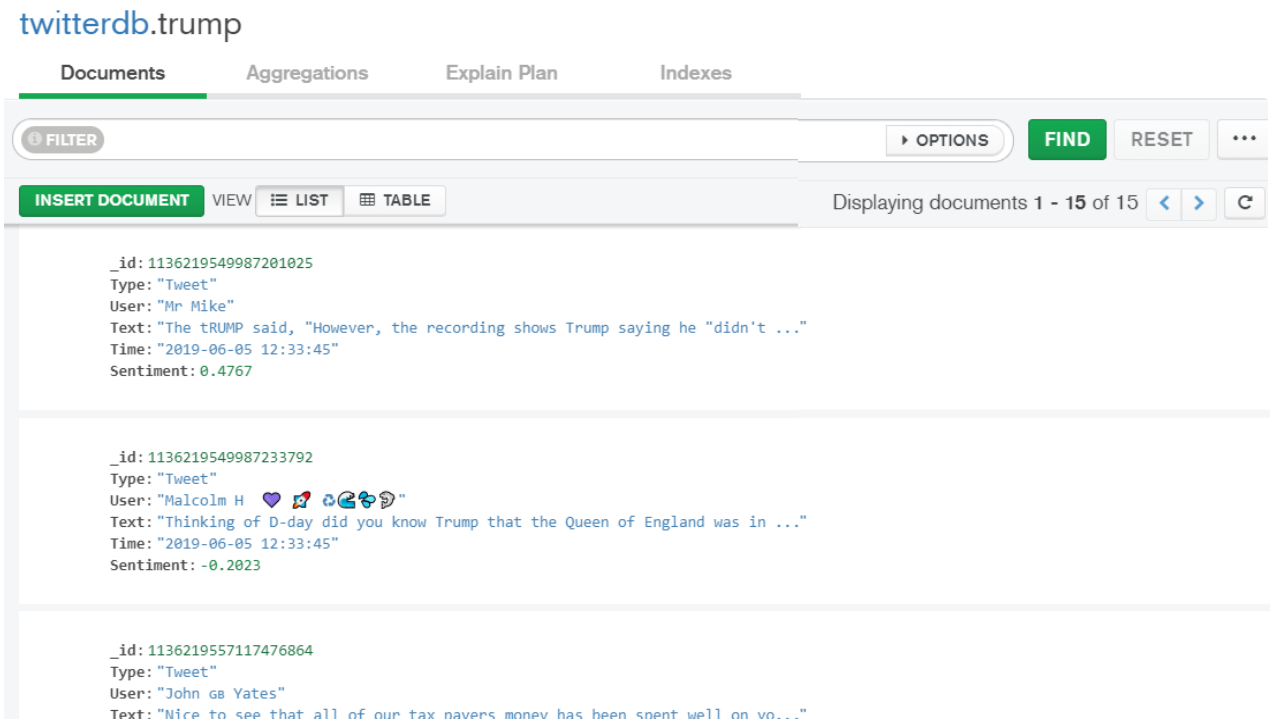

Ilustración 24: Datos almacenados en la BBDD

Capítulo 6: Resultados y evaluación

En este capítulo de la memoria se detallan las pruebas que han sido realizadas sobre nuestro sistema, comprobando el cumplimiento de los requisitos definidos en *Capítulo 3: Análisis del sistema*.

6.1 Pruebas unitarias

En este apartado se detallan las pruebas unitarias del sistema, que tienen como objetivo verificar que los requisitos descritos se cumplen y por tanto verificando la funcionalidad del sistema.

El formato tabular utilizado para describir las pruebas es el siguiente.

Identificador: P-XX	
Descripción	
Objetivo	
Resultado esperado Resultado obtenido	
RF/s cubierto/s	

Tabla 36: Plantilla de las pruebas unitarias

A continuación, se explica el significado de cada uno de los campos que componen la plantilla:

- **Identificador**. Código único de cada una de las pruebas. La nomenclatura utilizada será P-XX, donde:
  - P: Indica que es una prueba.
  - XX: Número de la prueba
- **Descripción**. Descripción breve pero detallada de la prueba.
- **Resultado esperado**. Resulta que se espera obtener tras la ejecución de la prueba.
- **Resultado obtenido**. Resuelto que se obtiene tras ejecutar la prueba, indicando si coincide con lo esperado o no.
- **RF/s cubierto/s**. Requisitos funcionales que han verificado su funcionamiento tras la ejecución de la prueba.

Las pruebas unitarias son las que se describen a continuación:

Identificador: P-01	
Descripción	Ejecución del código que se encarga de conectar con la API de búsqueda de Twitter. Se realizará una búsqueda desde el entorno de desarrollo.
Objetivo	Verificar que la conexión con Twitter es exitosa.
Resultado esperado	Se deberá imprimir por consola los tweets provenientes de la búsqueda.
Resultado obtenido	Se obtienen los tweets con todos sus campos incluidos y se muestran por consola.
RF/s cubierto/s	RF-01, RF-07

Tabla 37: Prueba unitaria P-01

Identificador: P-02	
Descripción	Ejecución del código que se encarga de conectar con la API de streaming de Twitter. Se realizará una búsqueda desde el entorno de desarrollo.
Objetivo	Verificar que la conexión con Twitter es exitosa.
Resultado esperado	Se deberá imprimir por consola los tweets en tiempo real.
Resultado obtenido	Se muestran tweets de forma continua por consola con todos sus campos
RF/s cubierto/s	RF-02, RF-05, RF-06

Tabla 38: Prueba unitaria P-02

Identificador: P-03	
Descripción	Ejecución del código que se encarga de analizar el texto de los tweets
Objetivo	Verificar que se obtiene un valor numérico que corresponde con el sentimiento
Resultado esperado	Se debe imprimir por consola el tweet con un campo más, que será el del sentimiento
Resultado obtenido	Se obtienen los tweets con los campos que se van a utilizar, en el que se incluye el sentimiento
RF/s cubierto/s	RF-04

Tabla 39: Prueba unitaria P-03

Identificador: P-04	
Descripción	Ejecución del código que se encarga de conectar con la base de datos para almacenar los tweets
Objetivo	Verificar que la conexión con MongoDB es exitosa y se encuentran los tweets almacenados
Resultado esperado	Deben aparecer los tweets en la aplicación de MongoDB
Resultado obtenido	Al refrescar la base de datos aparecen todos los tweets recogidos
RF/s cubierto/s	RF-03

Tabla 40: Prueba unitaria P-04

Identificador: P-05	
Descripción	Ejecución del código que se encarga de desplegar la aplicación web.
Objetivo	Comprobar que la aplicación web se ha lanzado.
Resultado esperado	Al entrar en la dirección IP que corresponde con el servidor local, ver que están las hojas de estilo cargadas y se imprime un mensaje de conexión por consola.
Resultado obtenido	Se accede a la ruta del servidor local y se observa que se han cargado las hojas de estilo, además de ver que se ha imprimido un mensaje de conexión por consola.
RF/s cubierto/s	RF-08

Tabla 41: Prueba unitaria P-05

Identificador: P-06	
Descripción	Ejecución del código que se encarga de generar el documento JSON.
Objetivo	Comprobar que se crea un documento JSON con los tweets incluidos dentro de él.
Resultado esperado	Documento JSON en las carpetas del sistema con los tweets.
Resultado obtenido	Tras ejecutar una búsqueda, aparece un documento JSON que contiene los datos obtenidos y que coinciden con los almacenados en la base de datos.
RF/s cubierto/s	RF-05

Tabla 42: Prueba unitaria P-06

Identificador: P-07	
Descripción	Intento de búsqueda o análisis sin rellenar todos los campos exigidos.
Objetivo	Obligar al usuario a que relleno todos los campos que se piden y no ejecutar la búsqueda o el análisis.
Resultado esperado	Cuadro señalando el campo que falta por completar.
Resultado obtenido	Al intentar ejecutar una búsqueda o un análisis sin rellenar todos los campos que se piden, aparece un cuadro en el campo vacío exigiendo que se rellene.
RF/s cubierto/s	RF-09, RF-10

Tabla 43: Prueba unitaria P-07

Identificador: P-08	
Descripción	Ejecución de diferentes búsquedas y análisis en tiempo real mientras el sistema está arrancado.
Objetivo	Hacer cuantas búsquedas y análisis en tiempo real se deseen y que no aparezcan excepciones del sistema
Resultado esperado	No encontrar ninguna excepción durante las búsquedas y análisis
Resultado obtenido	Se hace uso de las diferentes funcionalidades con total normalidad, visualizando los resultados
RF/s cubierto/s	RF-01, RF-02, RF-13, RF-14, RF-16, RF-17, RF-18, RF-19, RF-20, RF-21, RF-22

Tabla 44: Prueba unitaria P-08

Identificador: P-09	
Descripción	Ejecución de búsquedas de un mismo tema
Objetivo	Hacer cuantas búsquedas se desee de un mismo tema sin excepciones del sistema
Resultado esperado	Realizar búsquedas con normalidad y añadiéndose los resultados a los ya obtenidos con anterioridad
Resultado obtenido	Se añaden resultados a los anteriores y se pueden realizar cuantas se quiera sin excepciones
RF/s cubierto/s	RF-01, RF-12, RF-16, RF-17, RF-18, RF-19, RF-20, RF-22, RF-23

Tabla 45: Prueba unitaria P-09

Identificador: P-10	
Descripción	Descarga del documento JSON en la página de resultados
Objetivo	El usuario debe poder descargar el documento JSON que recoge todos los datos obtenidos y que se visualizan en la página de resultados
Resultado esperado	Descarga del documento con todos los datos
Resultado obtenido	Documento JSON con todos los tweets que se han visualizado en la página de resultados
RF/s cubierto/s	RF-15

Tabla 46: Prueba unitaria P-10

Identificador: P-11	
Descripción	Idioma de la información.
Objetivo	El usuario deberá visualizar los datos en el idioma que introduce en el campo que se establece para ello.
Resultado esperado	Tweets en el idioma seleccionado.
Resultado obtenido	Se obtiene la información en el idioma especificado en su campo.
RF/s cubierto/s	RF-11

Tabla 47: Prueba unitaria P-11

6.2 Matriz de trazabilidad

A continuación, se presenta la matriz de trazabilidad para el sistema, en la *Ilustración 25*. Con esta matriz se muestra cómo las pruebas realizadas cubren todos los requisitos funcionales que se definen para el sistema, asegurando un correcto funcionamiento.

RF/P	P-01	P-02	P-03	P-04	P-05	P-06	P-07	P-08	P-09	P-10	P-11
RF-01	X							X	X		
RF-02		X						X			
RF-03				X							
RF-04			X								
RF-05		X				X					
RF-06		X									
RF-07	X										
RF-08					X						
RF-09							X				
RF-10							X				
RF-11											X
RF-12									X		
RF-13								X			
RF-14								X			
RF-15										X	
RF-16								X	X		
RF-17								X	X		
RF-18								X	X		
RF-19								X	X		
RF-20								X	X		
RF-21								X			
RF-22								X	X		
RF-23									X		

Ilustración 25: Matriz de trazabilidad

Capítulo 7: Gestión del proyecto

En este capítulo se recogen aquellos aspectos relacionados con la gestión del proyecto. Se detallan aquí la metodología software que se ha seguido durante el desarrollo del proyecto, la planificación del trabajo y el presupuesto para el desarrollo, estimando costes de personal y equipo. También se incluye el impacto esperado a nivel socioeconómica, social y medioambiental.

7.1 Metodología software

La metodología software proporciona un marco de trabajo con el que poder planificar, estructurar y controlar todas las fases de desarrollo de nuestro sistema. Para este proyecto de fin de carrera se ha optado por un modelo que cuenta con distintas fases de análisis, diseño, implementación, pruebas y evaluación, donde siempre cabe la posibilidad de volver a una fase anterior si fuese preciso. En la *Ilustración 26* se muestran las fases del ciclo de vida de la metodología software en espiral seleccionada.

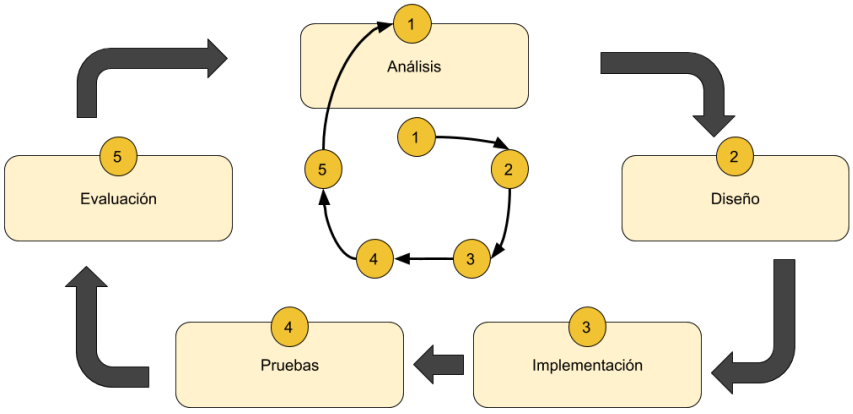

Ilustración 26: Fases de la metodología software

7.2 Planificación del proyecto

El proyecto tuvo como fecha de inicio el día 1 de marzo de 2019, y como fecha de finalización 1 de mayo de 2019, excluyendo la escritura de la memoria y revisiones, que fue hasta el día 5 de junio 2019.

Empleando la metodología en espiral que se describe en el apartado anterior, *7.1 Metodología software*, se realiza un único ciclo para realizar el desarrollo de MRT. Una vez se finaliza, completando todas las etapas correspondientes, se comienza con la memoria.

Trabajo	Fecha de inicio	Fecha de finalización	Duración (horas)
MRT	01/02/2019	01/05/2019	255
Escritura memoria TFG	01/02/2019	05/06/2019	57

Tabla 48: Planificación general del trabajo de fin de carrera

Una vez se ha indicado como es la planificación general del trabajo de fin de carrera, se detalla como ha sido la planificación para el desarrollo del sistema. Esta planificación se expone en la *Tabla 49*.

En la tabla se recoge cada fase del proyecto, y dentro de las fases, las tareas generales que se han realizado. De cada fase y de cada tarea se indica la duración en horas y la fecha aproximada de comienzo y de finalización.

Etapa	Fecha de inicio	Fecha de finalización	Duración (horas)
ANÁLISIS	**01/02/2019**	**20/03/2019**	**76**
Estado del arte	01/02/2019	28/02/2019	30
Investigación sobre herramientas de análisis de sentimientos	01/03/2019	04/03/2019	10
Investigación sobre funcionamiento de API de Twitter y librerías que trabajan con ella	04/03/2019	06/03/2019	12
Investigación sobre Websockets	06/03/2019	08/03/2019	8
Investigación sobre MongoDB	11/03/2019	11/03/2019	3
Especificación de requisitos	12/03/2019	14/03/2019	6
Revisión de requisitos	15/03/2019	15/03/2019	2
Definición de casos de uso	18/03/2019	19/03/2019	4
Revisión de casos de uso	20/03/2019	20/03/2019	1
DISEÑO	**21/03/2019**	**01/04/2019**	**37**
Estudio de viabilidad de diferentes alternativas	21/03/2019	25/03/2019	12
División del sistema en componentes	26/03/2019	28/03/2019	15
Interacción entre componentes	29/03/2019	01/04/2019	10

IMPLEMENTACIÓN	**02/04/2019**	**02/05/2019**	**122**
Desarrollo de la primera mitad de la aplicación web	02/04/2019	05/04/2019	18
Desarrollo del bloque de búsqueda	08/04/2019	16/04/2019	36
Desarrollo del bloque de análisis en tiempo real	17/04/2019	22/04/2019	20
Desarrollo del bloque de comunicación con la aplicación web	22/04/2019	26/04/2019	27
Desarrollo de la segunda mitad de la aplicación web	29/04/2019	02/05/2019	21
PRUEBAS	**03/05/2019**	**09/05/2019**	**20**
Definición de pruebas	03/05/2019	03/05/2019	5
Ejecución de pruebas	06/05/2019	09/05/2019	15

Tabla 49: Planificación para el desarrollo del sistema

Cabe destacar que durante los fines de semana no se trabajó en el proyecto, exceptuando el de la semana del 15 de abril.

Además de la planificación que se ha expuesto en la tabla, se procede a la realización de un diagrama de Gantt. Con él, se pretende mostrar gráficamente toda esta planificación. En este caso se medirá en días en vez de en horas, y aunque en la tabla se incluyen fines de semana, solo el de la semana del 15 de abril habrá sido utilizado como tiempo de trabajo.

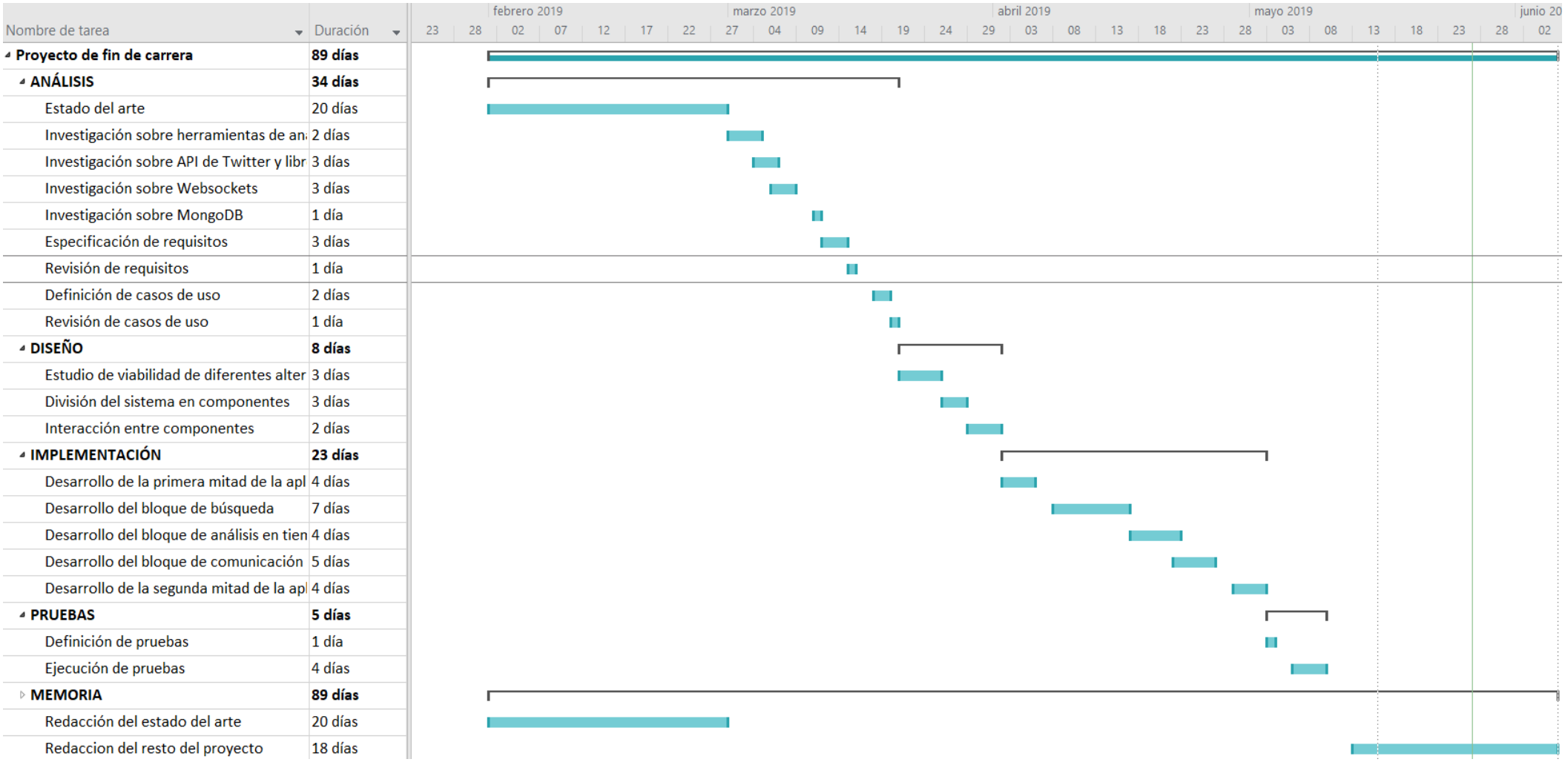

Ilustración 27: Diagrama de Gantt

7.3 Presupuesto

A continuación, se detalla el presupuesto que se necesitaría presentar para el desarrollo del proyecto que se ha llevado a cabo en el presente trabajo de fin de grado. Para ello, se calcularán los costes de personal, los costes materiales, y los costes indirectos. Por último, se realizará el cálculo del coste total.

7.3.1 Coste de personal

Los roles que participan en el proyecto son los siguientes:

- **Jefe de proyecto**. Persona encargada de coordinar y dirigir el proyecto.
- **Ingeniero principal**. Personal responsable de toda la investigación, análisis y diseño del proyecto.
- **Ingeniero desarrollador**. Persona encargada en última instancia de desarrollar y probar el sistema, asegurando su correcto funcionamiento.

Una vez se tienen los roles definidos, se procede al establecimiento de salarios y calcular el coste de cada uno de ellos dentro del proyecto, teniendo en cuenta el número de horas que ha desempeñado cada uno en el mismo. Los salarios se establecen a partir de la tabla salarial publicada en la resolución de la Dirección General de Empleo del 22 de febrero de 2018 por el BO, en el XVII Convenio colectivo estatal de empresas de consultoría y estudios de mercado y de opinión pública. El coste de personal se detalla en la siguiente tabla.

Rol	Coste por hora (€)	Horas de trabajo	Total (€)
Jefe de proyecto	29,74	23	684,02
Ingeniero principal	27,52	113	3109,76
Ingeniero desarrollador	20,87	142	2963,54
Total	-	278	**6757,32**

Tabla 50: Coste del personal del proyecto

7.3.2 Coste material

Por coste material se entiende el dinero invertido en software y hardware. En la *Tabla 51* se detallarán los recursos utilizados en el proyecto, así como su precio (no incluyendo aquellos que no han supuesto un desembolso económico), el periodo de amortización, el tiempo de uso, y el coste de cada uno de ellos en relación con las tres propiedades anteriores. Este coste es calculado mediante la siguiente ecuación:

$$Coste = \frac{precio}{periodo\ de\ amortización} * tiempo\ de\ uso$$

Producto	Precio (€)	Periodo de amortización (meses)	Tiempo de uso (meses)	Coste en proyecto (€)
PC	1305,59	48	3	81,59
Licencia Microsoft Office	299,00	48	3	18,69
Licencia Microsoft Project Standard 2019	849,00	48	3	53,06
Licencia PyCharm	199,00	12	3	49,75
Total	**203,09**			

Tabla 51: Coste material del proyecto

7.3.3 Coste total

Con los costes calculados en los anteriores apartados, nos queda por calcular los costes indirectos para así calcular el total del proyecto. Para calcular dichos costes, se consideran los siguientes:

- **Costes indirectos**. Se estima un 12% de costes indirectos. Dentro de este porcentaje se incluyen gastos como el acceso a internet o la electricidad.

- **Margen de riesgo**. Se estima un 18% de margen de riesgo. En este caso, el porcentaje cubre posibles imprevistos como averías en el equipo de desarrollo,

licencias de software que no se había contemplado, o cambio de las condiciones en la Api de Twitter.

- **Beneficio del proyecto**. Se estima un 30% de beneficio, que es lo que se llevaría la empresa con este proyecto

- **Impuestos**. Se añadirá un 21% del coste calculado hasta el momento, representando el IVA

En la *Tabla 52*, que se presenta a continuación, quedan recogidos todos los costes del proyecto, aplicando los porcentajes y mostrando el total.

Concepto	**Cantidad (€)**
Costes directos	6757,32
Costes indirectos (12%)	810,87
Margen de riesgo (18%)	1216,32
Beneficio del proyecto (30%)	2027,20
Base imponible	**10811,70**
IVA (21%)	2270,46
Coste total del proyecto	**13082,15**

Tabla 52: Coste total del proyecto

7.4 Plan de riesgos

En todos los sistemas informáticos se encuentran vulnerabilidades que pueden provocar situaciones en las que los proyectos en los que se trabaja puedan correr un riesgo mayor. El plan de riesgos identifica estas vulnerabilidades en los sistemas, para que una vez se han detectado, se tomen las medidas necesarias para prevenir los problemas que pudiesen surgir.

El plan de riesgos que se presenta en este trabajo de fin de carrera trata de proteger los sistemas críticos y los datos. La evaluación de riesgos incluye las siguientes actividades y acciones, tal y como se describen en la Norma ISO/IEC 27001:2005 de la familia de normas ISO 27000 para la seguridad informática:

- Identificación de activos.
- Identificación de los requisitos legales y de negocio relevantes para la identificación de activos.

- Valoración de activos identificados, teniendo en cuenta los requisitos legales y de negocio identificados en el punto anterior, y el impacto de una perdida de confidencialidad (C), disponibilidad (D) e integridad (I).
- Identificación de amenazas y vulnerabilidades para los activos identificados.
- Evaluación y cálculo de riesgo (alto, medio, bajo)

Cuando los riesgos se han identificado y evaluado se puedan tomar diferentes acciones. A continuación, se muestran las distintas posibilidades:

- **Mitigación del riesgo**. Mediante controles de seguridad.
- **Eliminación del riesgo**. Eliminando el activo en cuestión.
- **Traspaso del riesgo**. Haciendo que terceros se ocupen del riesgo en cuestión
- **Asumir el riesgo**. Determinar el nivel de riesgo como aceptable

En este caso, al ser utilizado un único equipo para el desarrollo del proyecto, se cuenta con un único riesgo existe, que seria de nivel alto. La avería del equipo, las filtraciones de datos personales o la corrupción del sistema de fichero serían las principales vulnerabilidades. El impacto del riesgo sería C D I (teniendo en cuenta los identificadores que se establecen anteriormente), y la acción a tomar en este caso seria la de mitigar el riesgo.

7.5 Impacto socioeconómico

El objetivo principal del proyecto es que las compañías encuentren en MRT una herramienta con la que entender mejor cuál es su clientela y qué pasos deberían seguir para mejorar su negocio. En un momento en el que la digitalización es la prioridad en las empresas de todos los sectores, MRT ofrece una solución barata y más precisa que muchas de las campañas y proyectos que se hayan podido llevar a cabo que tuviesen el mismo objetivo. Además, la información obtenida es útil para cualquier organización, sin importar la actividad que realice. Por tanto, el impacto socioeconómico que se busca con el trabajo de fin de carrera es la mejora en el aprovechamiento de los recursos por parte de las empresas, que como es lógico, piensan constantemente cómo mejorar sus productos y servicios teniendo en cuenta la opinión tanto de sus clientes más fieles, como de los que no hacen uso de ellos de manera tan frecuente. Este ahorro puede suponer en última instancia la creación de más puestos de trabajo por parte de la empresa, o incluso de mejoras salariales y laborales. Además, muchas organizaciones podrían originarse gracias a estas nuevas herramientas, convirtiéndose en consultoras especializadas en el tratamiento de datos.

Otro aspecto para tener en cuenta es el de los clientes. Muchas veces sienten que las empresas no escuchan sus quejas y que no ponen tierra de por medio para mejorar sus

productos y servicios. Gracias a MRT, los usuarios pueden encontrar en ella un altavoz donde estar seguros de que sus peticiones serán, al menos, escuchadas.

No todas las soluciones pueden estar dirigidas al sector privado. También en el público, los ayuntamientos pueden ver MRT como un instrumento con el que entender mejor las necesidades de los habitantes y, por tanto, facilitar y mejorar sus vidas.

MRT es una herramienta versátil y que con no mucha complejidad se puede adaptar a cualquier input escrito que se considere y que, por tanto, cuenta con un gran potencial que puede ser aún más grande, ya que hay mucho margen de mejora. Como en casi todas las cosas, el sistema puede ser utilizado para fines poco éticos, como puede ser aquel relacionado con la política. El compromiso con que MRT no se use para ello es total, y sólo se podrá utilizar con una aprobación que se otorgará tras un previo proceso de verificación de las actividades de la organización.

Capítulo 8: Conclusiones y trabajos futuros

En este último capítulo se detallan aquellas conclusiones extraídas de la realización del trabajo de fin de carrera. A continuación, se indicarán las diferentes conclusiones técnicas y líneas de trabajo futuro que se considera que se deben seguir a partir de este proyecto.

8.1 Conclusiones

La primera conclusión que se obtiene, y que es la más importante, es que los objetivos que se plantearon para el presente trabajo han sido cumplidos con éxito tras la realización de éste.

Se ha logrado construir un sistema, que mediante una aplicación web, nos da información sobre lo que piensan los usuarios en una red social, Twitter, respecto a una temática buscada. Además, se le da total libertad al usuario de nuestra plataforma para poder trabajar con los datos obtenidos en futuras investigaciones acerca del tema buscado. Otro objetivo que se ha cumplido es el relacionado con el diseño de la aplicación. Desde el principio, la experiencia de usuario ha sido el eje central del proyecto, en el que en ningún momento se ha querido descuidar, convirtiéndose en una prioridad. Se considera que en este ámbito el objetivo se ha cumplido exitosamente. Se cuenta con un diseño limpio, minimalista, pero sin carecer de funcionalidad, que intenta, dentro de la complejidad del mismo, facilitar el visionado de datos al usuario. Un aspecto también para tener en cuenta y relacionado con la experiencia de usuario con el que surgieron muchas dudas, fue la posibilidad de actualizar las gráficas sin tener que recargar constantemente la aplicación. Tras investigar y dar con las tecnologías necesarias, el objetivo se cumplió con creces, posibilitando al cliente ver los datos obtenidos en tiempo real.

El sistema tiene un tiempo de ejecución muy bajo para la cantidad de datos con los que se trabaja, respondiendo de forma rápida y eficaz a las peticiones por parte del usuario que se generan. Por tanto, se podría considerar seguir trabajando con el sistema para mejorarlo, ya que hay muchas cosas por implementar y perfeccionar, y que serán comentadas en el siguiente apartado.

Por último, hay que destacar la temática escogida para este trabajo de fin de carrera. Se trata de un proyecto interesante, basada en la demanda que existe hoy en día alrededor del tratado de datos, y que cumple las expectativas con lo que se considera un trabajo de fin de carrera en el grado de ingeniería informática.

8.2 Líneas futuras de trabajo

Este trabajo sin el análisis de sentimientos no tendría sentido. En ocasiones, si se estudia cada tweet por separado y se observa su sentimiento, se puede no estar de acuerdo con el valor que se aplica en este campo. Esto significa que hay un gran margen de mejora en el mundo del análisis de sentimiento, ya que no es sencillo, pues se debe tener en cuenta muchos factores. Puede que el tweet recogido venga precedido por otros, o que sea la respuesta a la publicación de otra persona. También puede que el texto tenga una intención irónica o sarcástica y no se identifique. El análisis nunca será certero si no se considera todo el contexto que rodea a un tweet, además de que también queda por perfeccionar detalles que corresponden con el análisis morfológico, sintáctico y semántico. No sería difícil incorporar otra herramienta de análisis de sentimientos si demostrase ser mejor que la actual, VaderSentiment, por lo que futuros desarrolladores deberían tener este aspecto en cuenta.

En lo relacionado con la visualización de datos, se considera que se podrían incorporar más gráficas que explicasen los resultados obtenidos. Se contempló, pero no se acabó implementando, una herramienta para mostrar las palabras más repetidas. Con un poco más de tiempo hubiera sido una funcionalidad más en el sistema, y que sin duda se acabará añadiendo en un futuro. Otras gráficas, como un mapamundi donde se mostrasen desde dónde se están recibiendo los tweets, en vez de la gráfica actual, o la posibilidad de ver cada tweet que entra en el análisis en tiempo real, se podrían contemplar, mejorando aún más la experiencia del usuario y exprimiendo todos los datos que nos ofrecen desde Twitter. Muchas son las limitaciones que se tienen por no hacer uso de su API de pago [37]. Si una compañía estuviese muy interesada en la herramienta, se sugeriría comprar el acceso a la API para empresas, con la que se tiene acceso a todos los tweets desde que existe la red social y por tanto las búsquedas serían mucho más significativas.

También se quiere destacar otra mejora importante, que es la de los idiomas. El proyecto sólo se ha desarrollado para información que se encuentra en inglés, pero si organizaciones que trabajan con idiomas diferentes encuentran en MRT una herramienta con la que obtener datos, el sistema se debería adaptar para poder ser utilizada con el nuevo idioma.

Por último, la posibilidad de tener la aplicación accesible para todo el mundo. Actualmente sólo puede ser usada en local, pero con herramientas como Google App Engine [38] se podría considerar lanzarla para que cualquiera pudiese utilizarla de manera responsable.

Como se puede observar, MRT cuenta con muchas mejoras que implicaría contar con una estrategia de negocio, pues cada una de ellas nos orienta hacía un ámbito que puede ser más privado o público. El sistema tiene mucho potencial y sin duda se considerará seguir mejorándola en un futuro próximo.

Referencias

[1] K. Smith, «98 estadísticas de las redes sociales para 2017 - Brandwatch | Brandwatch,» Mayo 2019. [En línea]. Disponible en: https://www.brandwatch.com/es/blog/116-estadisticas-de-las-redes-sociales/.

[2] Y. Roth y R. Johnson, «New developer requirements to protect our platform,» [En línea]. Disponible en: https://blog.twitter.com/developer/en_us/topics/tools/2018/new-developer-requirements-to-protect-our-platform.html. [Último acceso: Mayo 2019].

[3] C. F. Moise, *Personalization of Political Discoures On Social Media.,* Tesis, University of Bucharest, 2017 [En línea]. Disponible en: https://acl-bg.org/proceedings/2017/RANLP_W5%202017/

[4] Majoritas, "Majoritas | A political tech and data company," [En línea]. Disponible en:: https://majoritas.com/. [Último acceso: 13 Mayo 2019].

[5] A. Lorentz, "With Big Data, Context is a Big Issue," Marzo 2013. [En línea]. Disponible en:: https://www.wired.com/insights/2013/04/with-big-data-context-is-a-big-issue/. [Último acceso: 14 Mayo 2019].

[6] Deloitte Digital, «Segundo Estudio OCX: Transformando el negocio a través de la Voz del Cliente,» [En línea]. Disponible en:: https://www2.deloitte.com/es/es/pages/technology/articles/segundo-estudio-ocx.html. [Último acceso: 14 Mayo 2019].

[7] P. Carbonnelle, «PYPL PopularitY of Programming Language index,» [En línea]. Disponible en: http://pypl.github.io/PYPL.html. [Último acceso: 15 Mayo 2019].

[8] Jetbrains, «PyCharm: the Python IDE for Professional Developers by JetBrains,» [En línea]. Disponible en: https://www.jetbrains.com/pycharm/. [Último acceso: 15 Mayo 2019].

[9] Sublime Text, "Sublime Text - A sophisticated text editor for code, markup and prose," [En línea]. Disponible en: https://www.sublimetext.com/. [Último acceso: 15 Mayo 2019].

[10] I. Grand View Research, «Chatbot Market Size To Reach $1.25 Billion By 2025 | CAGR: 24.3%,» [En línea]. Disponible en: https://www.grandviewresearch.com/press-release/global-chatbot-market. [Último acceso: 4 Febrero 2019].

[11] R. Watcher, «Is NLP-Enabled Data Mining the Digital Breakthrough We've Been Waiting For?,» [En línea]. Disponible en: https://thehospitalleader.org/is-nlp-enabled-data-

mining-the-digital-breakthrough-weve-been-waiting-for/. [Último acceso: 4 Febrero 2019].

[12] K. Sennaar, «Machine Learning for Recruiting and Hiring – 6 Current Applications,» [En línea]. Disponible en: https://emerj.com/ai-sector-overviews/machine-learning-for-recruiting-and-hiring/. [Último acceso: 30 Mayo 2019].

[13] Vice, «Special Report - The Future of Work,» HBO, 19 Abril 2019. [En línea]. Disponible en: https://www.hbo.com/vice/special-reports/vice-special-report-the-future-of-work. [Último acceso: 26 Abril 2019].

[14] Lawgeex, «Platform - Lawgeex,» [En línea]. Disponible en: https://www.lawgeex.com/platform/. [Último acceso: 26 Abril 2019].

[15] C. Schneider, «The biggest data challenges that you might not even know you have,» [En línea]. Disponible en: https://www.ibm.com/blogs/watson/2016/05/biggest-data-challenges-might-not-even-know/. [Último acceso: 8 Febrero 2019].

[16] MonkeyLearn, «Sentiment Analysis: Nearly Everything You Need to Kow,» [En línea]. Disponible en: https://monkeylearn.com/sentiment-analysis/. [Último acceso: 7 Marzo 2019].

[17] R. Markey y F. Reichheld, «Introducing: The Net Promoter System®,» [En línea]. Disponible en: https://www.bain.com/insights/introducing-the-net-promoter-system-loyalty-insights/. [Último acceso: Abril 2019].

[18] McKinsey & Company, «McKinsey & Company | Global management consulting,» [En línea]. Disponible en: https://www.mckinsey.com/. [Último acceso: 12 Febrero 2019].

[19] McKinsey & Company, «Smart cities: Turning opportunity into reality,» [En línea]. Disponible en: https://www.mckinsey.com/~/media/mckinsey/industries/capital%20projects%20and%20infrastructure/our%20insights/voices%20on%20infrastructure%20turning%20the%20smart%20city%20opportunity%20into%20reality/voices-december-2017-web.ashx. [Último acceso: 12 Febrero 2019].

[20] La Vanguardia, «United Airlines protagoniza un nuevo incidente al expulsar violentamente a un pasajero de un avión,» [En línea]. Disponible en: https://www.lavanguardia.com/ocio/viajes/20170410/421605519191/united-airlines-nuevo-incidente-expulsion-violentamente-pasajero-avion.html. [Último acceso: 14 Febrero 2019].

[21] A. Marotti y L. Zumbach, «Video shows United Airlines' passenger dragged off plane,» [En línea]. Disponible en: https://www.latimes.com/business/ct-united-drags-passenger-0411-biz-20170410-story.html. [Último acceso: 14 Febrero 2019].

[22] Repustate, «Repustate: text analytics for businesses,» [En línea]. Disponible en: https://www.repustate.com/. [Último acceso: 18 Febrero 2019].

[23] MonkeyLearn, «MonkeyLearn - Text Analysis,» [En línea]. Disponible en: https://monkeylearn.com/. [Último acceso: 18 Febrero 2019].

[24] Block Six Analytics, «Block Six Analytics,» [En línea]. Disponible en: https://www.blocksixanalytics.com/. [Último acceso: 18 Febrero 2019].

[25] MeaningCloud, «Servicios web de analítica y minería de textos | MeaningCloud,» [En línea]. Disponible en: https://www.meaningcloud.com/es. [Último acceso: 18 Febrero 2019].

[26] NLTK Project, «Natural Language Toolkit -- NLTK 3.4.3 documentation,» [En línea]. Disponible en: https://www.nltk.org/. [Último acceso: 19 Febrero 2019].

[27] R. McGrath, «Twython -- wython 3.6.0 documentation,» [En línea]. Disponible en: https://twython.readthedocs.io/en/latest/. [Último acceso: Mayo 2019].

[28] C. Hutto y E. Gilbert, «GitHub - cjhutto/vaderSentiment: VADER Sentiment Analysis. VADER (Valence Aware Dictionary and sEntiment Reasoner) is a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media, and works well on,» [En línea]. Disponible en: https://github.com/cjhutto/vaderSentiment. [Último acceso: 2 Abril 2019].

[29] MongoDB, «The most popular database for modern apps | MongoDB,» [En línea]. Disponible en: https://www.mongodb.com/es. [Último acceso: Mayo 2019].

[30] A. Ronacher, «Welcome | Flask (A Python Microframework),» [En línea]. Disponible en: http://flask.pocoo.org/. [Último acceso: 22 Abril 2019].

[31] M. Brinkmann, «Flask-Bootstrap 3.3.7.1 documentation,» [En línea]. Disponible en: https://pythonhosted.org/Flask-Bootstrap/. [Último acceso: 25 Abril 2019].

[32] Bootstrap, «Bootstrap · The most popular HTML, CSS, and JS library in the world.,» [En línea]. Disponible en: https://getbootstrap.com/. [Último acceso: 2 Mayo 2019].

[33] Plotly, «Modern Analytic Apps for the Enterprise,» [En línea]. Disponible en: https://plot.ly/. [Último acceso: 14 Abril 2019].

[34] Twitter, «ntroduction to Tweet JSON — Twitter,» [En línea]. Disponible en: https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json. [Último acceso: 5 Marzo 2019].

[35] A. Rosen y I. Ihara , «Giving you more characters to express yourself,» [En línea]. Disponible en: https://blog.twitter.com/official/en_us/topics/product/2017/Giving-you-more-characters-to-express-yourself.html. [Último acceso: 15 Abril 2019].

[36] MongoDB, «PyMongo 3.8.0 Documentation,» [En línea]. Disponible en: https://api.mongodb.com/python/current/. [Último acceso: Mayo 2019].

[37] Twitter, «Overview - Twitter Developers,» [En línea]. Disponible en: https://developer.twitter.com/en/docs/tweets/search/overview. [Último acceso: 12 Abril 2019].

[38] Google, «App Engine - Crea backends web y móviles escalables en cualquier lenguaje,» [En línea]. Disponible en: https://cloud.google.com/appengine/. [Último acceso: 22 Mayo 2019].