ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

MÁSTER EN BIG DATA: TECNOLOGÍA Y ANALÍTICA AVANZADA

APLICACIÓN DE TÉCNICAS DE WEB SCRAPING Y PROCESAMIENTO DEL LENGUAJE NATURAL PARA LA EXTRACCIÓN Y EVALUACIÓN DE INFORMACIÓN DE UNA PÁGINA WEB DE EMPLEO

Autor: Guillermo Valle Gutiérrez

Director: Luis Pita-Romero Rodríguez

Madrid

Julio 2021

Declaro, bajo mi responsabilidad, que el Proyecto presentado con el título

Aplicación de técnicas web scraping y procesamiento del lenguaje natural para la extracción y evaluación de información de una página web de empleo

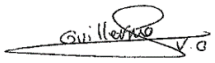
en la ETS de Ingeniería - ICAI de la Universidad Pontificia Comillas en el

curso académico 2020/21 es de mi autoría, original e inédito y

no ha sido presentado con anterioridad a otros efectos.

El Proyecto no es plagio de otro, ni total ni parcialmente y la información que ha sido

tomada de otros documentos está debidamente referenciada.



Fdo.: Guillermo Valle Gutiérrez Fecha: 06/07/2021

Autorizada la entrega del proyecto

EL DIRECTOR DEL PROYECTO

Fdo.: Luis Pita-Romero Rodríguez Fecha: 06/07/2021

Vº Bº del Coordinador de Proyectos

Fdo.: Carlos Morrás Ruiz-Falcó Fecha: ……/ ……/ ……

AUTORIZACIÓN PARA LA DIGITALIZACIÓN, DEPÓSITO Y DIVULGACIÓN EN RED DE PROYECTOS FIN DE GRADO, FIN DE MÁSTER, TESINAS O MEMORIAS DE BACHILLERATO

1º. Declaración de la autoría y acreditación de la misma.

El autor D. Guillermo Valle Gutiérrez DECLARA ser el titular de los derechos de propiedad intelectual de la obra: *Aplicación de técnicas web scraping y procesamiento del lenguaje natural para la extracción y evaluación de información de una página web de empleo*, que ésta es una obra original, y que ostenta la condición de autor en el sentido que otorga la Ley de Propiedad Intelectual.

2º. Objeto y fines de la cesión.

Con el fin de dar la máxima difusión a la obra citada a través del Repositorio institucional de la Universidad, el autor **CEDE** a la Universidad Pontificia Comillas, de forma gratuita y no exclusiva, por el máximo plazo legal y con ámbito universal, los derechos de digitalización, de archivo, de reproducción, de distribución y de comunicación pública, incluido el derecho de puesta a disposición electrónica, tal y como se describen en la Ley de Propiedad Intelectual. El derecho de transformación se cede a los únicos efectos de lo dispuesto en la letra a) del apartado siguiente.

3º. Condiciones de la cesión y acceso

Sin perjuicio de la titularidad de la obra, que sigue correspondiendo a su autor, la cesión de derechos contemplada en esta licencia habilita para:

a) Transformarla con el fin de adaptarla a cualquier tecnología que permita incorporarla a internet y hacerla accesible; incorporar metadatos para realizar el registro de la obra e incorporar "marcas de agua" o cualquier otro sistema de seguridad o de protección.
b) Reproducirla en un soporte digital para su incorporación a una base de datos electrónica, incluyendo el derecho de reproducir y almacenar la obra en servidores, a los efectos de garantizar su seguridad, conservación y preservar el formato.
c) Comunicarla, por defecto, a través de un archivo institucional abierto, accesible de modo libre y gratuito a través de internet.
d) Cualquier otra forma de acceso (restringido, embargado, cerrado) deberá solicitarse expresamente y obedecer a causas justificadas.
e) Asignar por defecto a estos trabajos una licencia Creative Commons.
f) Asignar por defecto a estos trabajos un HANDLE (URL *persistente)*.

4º. Derechos del autor.

El autor, en tanto que titular de una obra tiene derecho a:

a) Que la Universidad identifique claramente su nombre como autor de la misma
b) Comunicar y dar publicidad a la obra en la versión que ceda y en otras posteriores a través de cualquier medio.
c) Solicitar la retirada de la obra del repositorio por causa justificada.
d) Recibir notificación fehaciente de cualquier reclamación que puedan formular terceras personas en relación con la obra y, en particular, de reclamaciones relativas a los derechos de propiedad intelectual sobre ella.

5º. Deberes del autor.

- El autor se compromete a:

a) Garantizar que el compromiso que adquiere mediante el presente escrito no infringe ningún derecho de terceros, ya sean de propiedad industrial, intelectual o cualquier otro.
b) Garantizar que el contenido de las obras no atenta contra los derechos al honor, a la intimidad y a la imagen de terceros.
c) Asumir toda reclamación o responsabilidad, incluyendo las indemnizaciones por daños, que pudieran ejercitarse contra la Universidad por terceros que vieran infringidos sus derechos e intereses a causa de la cesión.
d) Asumir la responsabilidad en el caso de que las instituciones fueran condenadas por infracción

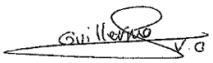
de derechos derivada de las obras objeto de la cesión.

6º. Fines y funcionamiento del Repositorio Institucional.
La obra se pondrá a disposición de los usuarios para que hagan de ella un uso justo y respetuoso con los derechos del autor, según lo permitido por la legislación aplicable, y con fines de estudio, investigación, o cualquier otro fin lícito. Con dicha finalidad, la Universidad asume los siguientes deberes y se reserva las siguientes facultades:

- La Universidad informará a los usuarios del archivo sobre los usos permitidos, y no garantiza ni asume responsabilidad alguna por otras formas en que los usuarios hagan un uso posterior de las obras no conforme con la legislación vigente. El uso posterior, más allá de la copia privada, requerirá que se cite la fuente y se reconozca la autoría, que no se obtenga beneficio comercial, y que no se realicen obras derivadas.
- La Universidad no revisará el contenido de las obras, que en todo caso permanecerá bajo la responsabilidad exclusive del autor y no estará obligada a ejercitar acciones legales en nombre del autor en el supuesto de infracciones a derechos de propiedad intelectual derivados del depósito y archivo de las obras. El autor renuncia a cualquier reclamación frente a la Universidad por las formas no ajustadas a la legislación vigente en que los usuarios hagan uso de las obras.
- La Universidad adoptará las medidas necesarias para la preservación de la obra en un futuro.
- La Universidad se reserva la facultad de retirar la obra, previa notificación al autor, en supuestos suficientemente justificados, o en caso de reclamaciones de terceros.

Madrid, a 06 de julio de 2021.

ACEPTA

Fdo Guillermo Valle Gutiérrez 

Motivos para solicitar el acceso restringido, cerrado o embargado del trabajo en el Repositorio Institucional:

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA (ICAI)

MÁSTER EN BIG DATA: TECNOLOGÍA Y ANALÍTICA AVANZADA

APLICACIÓN DE TÉCNICAS DE WEB SCRAPING Y PROCESAMIENTO DEL LENGUAJE NATURAL PARA LA EXTRACCIÓN Y EVALUACIÓN DE INFORMACIÓN DE UNA PÁGINA WEB DE EMPLEO

Autor: Guillermo Valle Gutiérrez

Director: Luis Pita-Romero Rodríguez

Madrid

Julio 2021

Agradecimientos

Quiero expresar mi agradecimiento sobre todo a mis padres, Santos y Juana, y a mi hermana María, por la educación recibida a lo largo de mi vida. También a Clara por el apoyo incondicional que me ha dado siempre. Agradecer también a todas las personas que he conocido en Madrid, en el C.M.U Barberán y Collar; mis compañeros en la UPM y en ICAI; las que he conocido como jugador de fútbol en el Chamberí FS; y las que he conocido en mi vida laboral en KPMG, The Cocktail y Baobab, por haber hecho que estos años vividos en Madrid hayan sido de los mejores de mi vida.

Gracias,

Guillermo

APLICACIÓN DE TÉCNICAS DE WEB SCRAPING Y PROCESAMIENTO DEL LENGUAJE NATURAL PARA LA EXTRACCIÓN Y EVALUACIÓN DE INFORMACIÓN DE UNA PÁGINA WEB DE EMPLEO

Autor: Valle Gutiérrez, Guillermo.
Director: Pita-Romero Rodríguez, Luis.
Entidad Colaboradora: Baobab Soluciones SL.

RESUMEN DEL PROYECTO

El presente trabajo realiza un estudio de la viabilidad de la aplicación de técnicas de procesamiento de lenguaje natural para el análisis de sentimiento en valoraciones. Para ello, ha sido necesaria la implementación de un módulo de *web scraping* y el desarrollo de modelos regresión logística y Naive Bayes para la clasificación de texto basado en el análisis de sentimiento. Además, se ha realizado una comparación de los resultados de los modelos desarrollados con dos modelos pre-entrenadas de las librerías NLTK y TextBlob. Tras el desarrollo del proyecto se han obtenido resultados satisfactorios en los modelos de Naive Bayes.

Palabras clave: *web scraping*, procesamiento del lenguaje natural, Python, NLTK, BeautifulSoup, Selenium, VADER, TextBlob.

1. Introducción

El procesamiento del lenguaje natural o NLP, expresado así por sus siglas en inglés, es el campo de la inteligencia artificial que permite a las máquinas ser capaz de leer, entender y derivar significados del lenguaje humano.

Esta disciplina se encuentra entre el mundo de la ciencia de datos y el lenguaje humano y se está expandiendo a muchas industrias. Pero no sería posible si no se tuviese la capacidad de computación y la gran cantidad de información que se tiene hoy en día.

En relación con estas grandes cantidades de datos cabe destacar los dos tipos principales de fuentes de datos en una compañía: las fuentes de datos internas y externas. Las fuentes de datos internas son aquellos datos de los que dispone la empresa, que se encuentran alojados en sus sistemas de información, sus servidores y bases de datos. Mientras que las fuentes de datos externas son aquellas de las que la empresa no dispone en el momento actual, pero puede recurrir a ellas para obtener información. En el presente proyecto se ha utilizado Internet como fuente de datos externa, y la forma utilizada para acceder a la información de manera automatizada ha sido la conocida como *web scraping*.

El trabajo realizado se ha centrado en el **desarrollo de modelos de clasificación en base a los sentimientos**. Para ello, se ha desarrollado un módulo de ***web scraping* para la extracción de valoraciones de una página web de empleo**, y mediante **técnicas de procesamiento del lenguaje natural** se han **desarrollado los modelos que permiten clasificar nuevas valoraciones como valoraciones positivas o negativas**.

2. Definición del proyecto

El objetivo principal del proyecto es el **análisis de viabilidad de la aplicación de técnicas de procesamiento del lenguaje natural para el análisis de sentimiento en valoraciones sobre empleos**. Para el correcto cumplimiento del objetivo principal ha sido necesaria la definición de unos objetivos específicos. El primero de ellos ha sido la implementación de un módulo de *web scraping* para la extracción de las valoraciones de una página web de empleo. El segundo ha sido el desarrollo de modelos de clasificación mediante el uso de técnicas de procesamiento del lenguaje natural. El tercero y último ha sido la exploración de las aplicaciones de estos modelos en contextos empresariales.

El proyecto se ha dividido en **dos etapas principales**: el desarrollo de módulo de *web scraping* para la extracción de las valoraciones de la página web, y el desarrollo de los modelos de clasificación de las valoraciones extraídas.

El **módulo de *web scraping*** se ha dividido a su vez en tres fases. Una primera fase de obtención de información general, continuando con una segunda fase de extracción de toda la información de las valoraciones y terminando con una fase de limpieza y estructuración de los datos extraídos.

Para el **desarrollo de los modelos de clasificación**, la metodología empleada consiste en realizar en la primera fase un procesado de las valoraciones mediante el uso de técnicas del procesamiento del lenguaje natural. Posteriormente, se realiza una transformación de la información procesada en variables aprovechables por los modelos. Por último, se construyen los modelos de clasificación, se realiza su entrenamiento con un conjunto de entrenamiento y se evalúan los resultados sobre un conjunto de test.

3. Descripción del módulo de web scraping

Para el desarrollo del módulo de *web scraping*, se ha realizado un **estudio de las diferentes páginas web de empleos** que contienen la información necesaria para el cumplimiento del proyecto. Antes de la implementación del módulo de *web scraping*, se llevó a cabo un **análisis de la legalidad** de obtener esa información para el fin del proyecto en concreto. Una vez decidió la página web y se validó la legalidad, se realizó un **estudio de las implementaciones posibles a desarrollar** para extraer esa información.

Para el presente proyecto, se desarrollaron **dos implementaciones** que extraen la información necesaria para el proyecto. La primera de ellas está basada en la **programación HTTP de los servidores web**, de tal forma que realiza las mismas peticiones que el navegador y extrae la información de la respuesta de la petición. La segunda se basa en la **herramienta Selenium**, que permite el control de un navegador web. Esta herramienta fue creada inicialmente para la realización de pruebas de las páginas web en desarrollo, pero al controlar un navegador web, se ha convertido en una herramienta muy utilizada en el desarrollo de módulo de *web scraping*.

Ambas implementaciones están divididas en **tres fases**: La primera fase ha consistido en la extracción de empresas registradas en la página web de empleo, almacenando el número de valoraciones y la dirección URL dónde encontrar dichas valoraciones. En la

segunda fase se ha implementado el módulo de extracción de las valoraciones de cualquier empresa registrada en la página web y en la tercera se ha realizado una limpieza de la información extraída y la separación por el idioma del texto de la valoración.

Los desarrollos correspondientes a esta parte se han llevado a cabo utilizando el software libre **Python**, seleccionado principalmente por el amplio número de librerías de las que dispone. Concretamente se han utilizado **BeautifulSoup** y **Selenium**.

4. Descripción de los modelos de clasificación

El desarrollo de los modelos de clasificación se centra en el **modelo de regresión logística** y en el **modelo de clasificación de Naive Bayes**. Ambos consisten en un tratamiento inicial de los datos, la extracción de las variables aprovechables por los modelos, la construcción de los modelos, y el entrenamiento y validación de los modelos.

El conjunto de datos utilizado para el desarrollo de los modelos de clasificación en base a los sentimientos utiliza la nota de la valoración para realizar una clasificación binaria (positiva o negativa). Las valoraciones de puntuación 1-2 han sido clasificadas como malas valoraciones y las de puntuación 4-5 han sido clasificadas como buenas. Hay que destacar que el conjunto de valoraciones extraído está desbalanceado, ya que tiene un 75% de valoraciones positivas y un 25% de valoraciones negativas.

Para garantizar un análisis de los modelos y sus predicciones adecuado, se ha dividido el conjunto de datos en datos de entrenamiento y datos de test. Los datos de test han sido aislados de la construcción de los modelos y para así poder comparar las previsiones del modelo con dichos valores completamente independientes.

Además de la construcción de los modelos de clasificación, se llevaron a cabo predicciones sobre el conjunto de test con dos modelos pre-entrenados de las librerías NLTK (*Natural Language Toolkit*), concretamente VADER, y TextBlob. Estas predicciones han servido para realizar una comparación de los modelos implementados con librerías entrenadas específicamente en el análisis de sentimientos.

Del mismo modo que el módulo de *web scraping*, el desarrollo de los modelos se llevó a cabo utilizando el software libre **Python**, seleccionado principalmente por el amplio número de librerías relacionadas con el NLP disponibles. Concretamente se han utilizado **NLTK** y **TextBlob**.

En la Ilustración 1 se muestra el diagrama de flujo desarrollado en el presente proyecto.

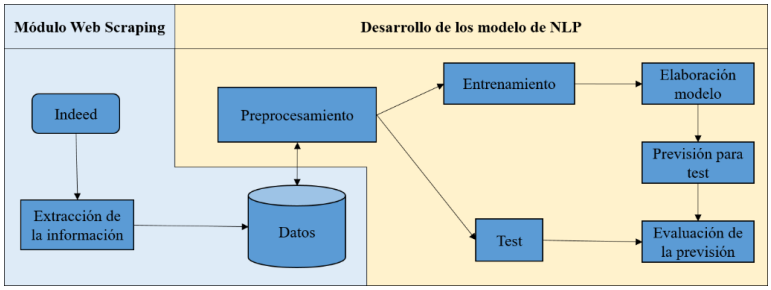


Ilustración 1. Diagrama de flujo del proyecto.

5. Resultados

Tras la construcción de los modelos de clasificación en base a los sentimientos de las valoraciones, se realizaron las pruebas de los distintos modelos. Los modelos desarrollados han sido probados con distintos tipos de información extraída de las valoraciones. Inicialmente, solo se introdujo el **texto de la valoración**. En posteriores iteraciones se añadieron los **textos correspondientes al título de la valoración** y **a las ventajas y desventajas** que los usuarios han descrito de las empresas. Los mejores resultados se obtienen con el modelo de **Naive Bayes**, ya que utiliza la probabilidad a priori que permite obtener mejores resultados en conjuntos desbalanceados. El conjunto de información con mejores resultados ha sido el que contiene la información del texto, título, ventajas y desventajas de la valoración. En la Tabla 1 se muestran las métricas obtenidas con el modelo de Naive Bayes para este conjunto en concreto.

Modelo	Precision	Recall	Accuracy	F1 score
Naive Bayes	0.9306	0.9267	0.8858	0.9286

Tabla 1. Resultados obtenidos con Naive Bayes.

6. Conclusiones

La realización del trabajo descrito permite validar la **viabilidad de la aplicación de técnicas de procesamiento de lenguaje natural para el análisis de sentimiento en valoraciones**. De este modo se puede afirmar que la aplicación de las técnicas de NLP es de utilidad para la identificación de sentimientos en las valoraciones extraídas.

En particular, de los resultados obtenidos se concluye que los **modelos de Naive Bayes implementados son adecuados para llevar a cabo la clasificación de valoraciones como positivas o negativas**. Además, durante la realización del trabajo se ha obtenido un amplio conocimiento en las técnicas de *web scraping*, un desarrollo personal y profesional, además de todo el componente teórico asociado al proyecto.

7. Referencias

[1] David Zimbra, A. A. (2018). The State-of-the-Art in Twitter Sentiment Analysis: A Review and Benchmark Evaluation. 29.

[2] Hanhoon Kang, S. J. (2012). Senti-lexicon and improved Naïve Bayes algorithms for sentiment analysis of restaurant reviews. 11.

[3] Lopamudra Dey, S. C. (2016). Sentiment Analysis of Review Datasets using Naïve Bayes' and K-NN Classifier. 7.

[4] Martin., D. J. (2020). Logistic Regression. Speech and Language Processing., 21. Retrieved from https://web.stanford.edu/~jurafsky/slp3/5.pdf

[5] Monisha Kanakaraj, R. M. (2015). Performance Analysis of Ensemble Methods on Twitter Sentiment Analysis using NLP Techniques. 2.

[6] Nadia Felix, E. R. (2014). Tweet Sentiment Analysis with Classifier Ensembles. 31.

[7] Peng Shi, W. Y. (2020). Logistic Regression for Sentiment Analysis on Large Scale Social Media Posts via Apache Spark. 6.

[8] Prashant Johri, M. K.-T. (2020). Natural Language Processing: History, Evolution, Application and Future Work. 8.

[9] Raschka, S. (2014). Naive Bayes and Text Classification I. 20.

[10] Vikas Khullar, A. P. (2017). Sentiment classification on big data using Naïve bayes and logistic regression. 6.

APLICACIÓN DE TÉCNICAS DE WEB SCRAPING Y PROCESAMIENTO DEL LENGUAJE NATURAL PARA LA EXTRACCIÓN Y EVALUACIÓN DE INFORMACIÓN DE UNA PÁGINA WEB DE EMPLEO

Author: Valle Gutiérrez, Guillermo.
Supervisor: Pita-Romero Rodríguez, Luis.
Collaborating Entity: Baobab Soluciones SL.

ABSTRACT

This paper studies the feasibility of the application of natural language processing techniques for sentiment analysis in ratings. For this purpose, it has been necessary the implementation of a web scraping module and the development of logistic regression and Naive Bayes models for text classification based on sentiment analysis. In addition, a comparison of the results of the developed models with two pre-trained models from the NLTK and TextBlob libraries has been carried out. After the development of the project, satisfactory results have been obtained for the Naive Bayes models.

Keywords: web scraping, natural language processing, Python, NLTK, BeautifulSoup, Selenium, VADER, TextBlob.

1. Introduction

Natural language processing, or NLP, is the field of artificial intelligence that enables machines to be able to read, understand and derive meaning from human language.

This discipline lies between the world of data science and human language and is expanding into many industries. But it would not be possible without the computing power and the vast amounts of information available today.

In relation to these large amounts of data, there are two main types of data sources in a company: internal data sources and external data sources. Internal data sources are those available to the company, which are housed in its information systems, servers, and databases. While external data sources are those that the company does not have at the present time but can be used to obtain information. In this project, the Internet has been used as an external data source, and the way used to access the information in an automated way has been known as web scraping.

The work done has focused on the **development of sentiment-based classification models. For this purpose, a web scraping module has been developed for the extraction of ratings from an employment web page, and by means of natural language processing techniques, models have been developed to classify new ratings as positive or negative.**

2. Project definition

The main objective of the project is the **feasibility analysis of the application of natural language processing techniques for sentiment analysis in job evaluations**. For the correct fulfilment of the main goal, it has been necessary to define some specific

objectives. The first of them has been the implementation of a web scraping module for the extraction of ratings from an employment web page. The second was the development of classification models using natural language processing techniques. The third and last one has been the exploration of the applications of these models in business contexts.

The project has been divided into two main stages: the development of the web scraping module for the extraction of the ratings from the web page, and the development of the classification models of the extracted ratings.

The web scraping module has been divided into three phases. A first phase of obtaining general information, continuing with a second phase of extraction of all the information from the ratings and finishing with a phase of cleaning and structuring the extracted data.

For **the development of the classification models**, the methodology used consists of processing the ratings in the first phase using natural language processing techniques. Subsequently, the processed information is transformed into variables that can be used by the models. Finally, the classification models are built, trained with a training set and the results are evaluated on a test set.

3. Description of the web scraping module

For the development of the web scraping module, **a study of the different employment web pages** that contain the necessary information for the fulfilment of the project was carried out. Before the implementation of the web scraping module, **an analysis of the legality** of obtaining that information for the purpose of the specific project was carried out. Once the web page was decided and the legality was validated, a **study of the possible implementations to be developed** to extract that information was carried out.

For the present project, **two implementations** were developed to extract the necessary information for the project. The first one is based on **the HTTP programming** of the web servers, in such a way that it performs the same requests as the browser and extracts the information from the response to the request. The second is based on the **Selenium tool**, which allows the control of a web browser. This tool was initially created for testing web pages under development, but by controlling a web browser, it has become a widely used tool in the development of web scraping modules.

Both implementations are divided into **three phases**: the first phase has consisted in the extraction of companies registered on the employment web page, storing the number of ratings and the URL address where to find those ratings. In the second phase, the module for extracting the reviews of any company registered on the website was implemented, and in the third phase, the extracted information was cleaned, and the text of the review was separated by language.

The developments corresponding to this part have been carried out using the free software **Python**, selected mainly because for its large number of libraries available. Specifically, **BeautifulSoup** and **Selenium** were used.

4. Description of the classification models

The development of the classification models focuses on **the logistic regression model** and the **Naive Bayes classification model**. Both consist of initial data processing, extraction of variables exploitable by the models, model building, and model training and validation.

The dataset used for the development of the sentiment-based classification models uses the rating score to perform a binary classification (positive or negative). Ratings scored 1-2 have been classified as bad ratings and ratings scored 4-5 have been classified as good. It should be noted that the extracted set of ratings is unbalanced, as it has 75% positive ratings and 25% negative ratings.

To ensure a proper analysis of the models and their predictions, the data set has been divided into training data and test data. The test data has been isolated from the construction of the models to be able to compare the model predictions with these completely independent values.

In addition to the construction of the classification models, predictions on the test set were carried out with two pre-trained models from the NLTK (Natural Language Toolkit) libraries, namely VADER, and TextBlob. These predictions have been used to make a comparison of the models implemented with libraries specifically trained in sentiment analysis.

Like the web scraping module, the development of the models was carried out using the free software **Python**, selected mainly because of the large number of NLP-related libraries available. Specifically, **NLTK** and **TextBlob** were used.

Figure 1 shows the flow diagram developed in this project.

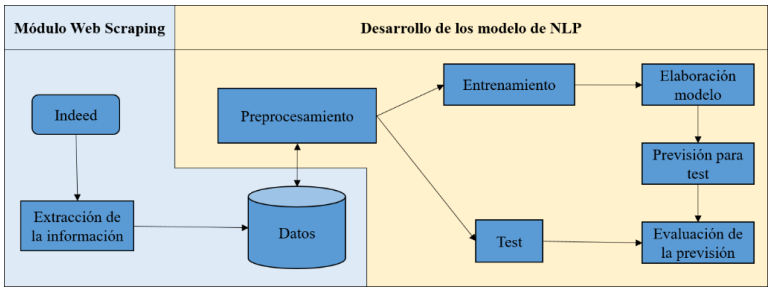


Figure 1. Flow diagram.

5. Results

After the construction of the classification models based on the sentiments of the ratings, the different models were tested. The developed models were tested with different types of information extracted from the ratings. Initially, **only the text of the valuation** was introduced. In later iterations, **the texts corresponding to the title of the valuation and the advantages and disadvantages** that the users have described of the companies were added. The best results are obtained with the **Naive Bayes** model, since it uses the a priori probability that allows obtaining better results in unbalanced sets. The information

set with the best results has been the one containing the information of the text, title, advantages, and disadvantages of the valuation. Table 1 shows the metrics obtained with the Naive Bayes model for this set.

Model	Precision	Recall	Accuracy	F1 score
Naive Bayes	0.9306	0.9267	0.8858	0.9286

Table 1. Results.

6. Conclusions

The work described above allows validating the **feasibility of the application of natural language processing techniques for sentiment analysis in ratings**. Thus, it can be confirmed that the application of NLP techniques is useful for the identification of sentiment in the extracted ratings.

From the obtained results it is concluded that the implemented **Naive Bayes models are suitable to carry out the classification of ratings as positive or negative**. In addition, during the realization of the work, a wide knowledge in web scraping techniques, a personal and professional development, as well as all the theoretical component associated with the project have been obtained.

7. References

[1] David Zimbra, A. A. (2018). The State-of-the-Art in Twitter Sentiment Analysis: A Review and Benchmark Evaluation. 29.

[2] Hanhoon Kang, S. J. (2012). Senti-lexicon and improved Naïve Bayes algorithms for sentiment analysis of restaurant reviews. 11.

[3] Lopamudra Dey, S. C. (2016). Sentiment Analysis of Review Datasets using Naïve Bayes' and K-NN Classifier. 7.

[4] Martin., D. J. (2020). Logistic Regression. Speech and Language Processing., 21. Retrieved from https://web.stanford.edu/~jurafsky/slp3/5.pdf

[5] Monisha Kanakaraj, R. M. (2015). Performance Analysis of Ensemble Methods on Twitter Sentiment Analysis using NLP Techniques. 2.

[6] Nadia Felix, E. R. (2014). Tweet Sentiment Analysis with Classifier Ensembles. 31.

[7] Peng Shi, W. Y. (2020). Logistic Regression for Sentiment Analysis on Large Scale Social Media Posts via Apache Spark. 6.

[8] Prashant Johri, M. K.-T. (2020). Natural Language Processing: History, Evolution, Application and Future Work. 8.

[9] Raschka, S. (2014). Naive Bayes and Text Classification I. 20.

[10] Vikas Khullar, A. P. (2017). Sentiment classification on big data using Naïve bayes and logistic regression. 6.

Índice de la memoria

Índice de figuras

Índice de tablas

Capítulo 1. INTRODUCCIÓN

El procesamiento del lenguaje natural o NLP, expresado así por sus siglas en inglés, es el campo de la inteligencia artificial (AI) que permite a las máquinas ser capaz de leer, entender y derivar significados del lenguaje humano.

Todo lo que el ser humano es capaz de expresar (referido al lenguaje: escrito o hablado) contiene gran cantidad de información. El tema elegido, el tono, la selección de las palabras, la finalidad, todo lo que lo envuelve tiene infinidad de información y valor que pueden ser extraídos.

El problema es que toda esta información no tiene un patrón común, es decir, forma parte de lo que se denominan datos desestructurados. Todo lo que ser humano es capaz de producir (declaraciones, artículos, libros, incluso tweets o valoraciones) son datos que no se pueden organizar de la forma tradicional de una base de datos relacional (filas y columnas).

Pero gracias a la potencia de computación actual se pueden aprovechar algunas técnicas que permiten la extracción de información del texto o la interpretación desde otro tipo de enfoque. Este tipo de técnicas son las de procesamiento del lenguaje natural, y su enfoque más sencillo es la extracción de palabras clave para analizar el contexto del documento o texto que se esté procesando, pero se puede llegar a entender el significado detrás de cada una de las palabras (lingüística cognitiva) y los sentimientos expresados.

Esta disciplina se encuentra entre el mundo de la ciencia de datos y el lenguaje humano y se está expandiendo a muchas industrias. Pero no sería posible si no se tuviese la capacidad de computación y la gran cantidad de información que se tiene hoy en día. Hay multitud de fuentes de información, desde las bases de datos y documentos físicos tradicionales, hasta el Internet de las Cosas (IoT) e Internet.

Existen dos tipos de información en el ambiente empresarial: las fuentes internas y las fuentes externas. Se diferencian en que las fuentes de datos externas son aquellas que no han sido recogidas o que no dispone la empresa y las fuentes de datos internas son datos de los que dispone la empresa, que se encuentran en sus sistemas de información, los servidores y bases de datos (incluso documentos físicos que se están pasando a digital en los últimos años). Existen muchos tipos de fuentes de datos externas, una de las más conocidas y usadas dentro de las empresas son las redes sociales. La información de las redes sociales puede ayudar a conocer mejor a los clientes y clientes potenciales, e incluso conectar con ellos.

En cuanto a la información externa que se puede conseguir a través de Internet, la técnica o forma de hacerlo de una manera automática es conocida como *web scraping*. Consiste en la extracción de datos de páginas web. Este tipo de técnicas son útiles cuando no se dispone de la información necesaria a través de descarga de ficheros de una página web, o a través de una API[1].

El trabajo realizado se ha centrado en la **aplicación de técnicas de *web scraping* para la extracción de datos de una página web de empleo**, y en el **uso de técnicas de procesamiento del lenguaje natural para la clasificación de la información extraída**.

1.1 MOTIVACIÓN DEL PROYECTO

El objetivo principal de las empresas que trabajan con datos es tener una gran cantidad de información de calidad, para poder analizarla y tomar las decisiones que beneficien a la empresa. Si no se dispone de la información necesaria para poder realizar este tipo de proyectos, se tiene que recurrir a la recopilación de información mediante fuentes externas.

Otro de los objetivos es poder identificar los problemas tanto internos como externos para adelantarse a la materialización de estos, evitando así las consecuencias negativas que

[1] Acrónimo de *Application Programming Interface*, se define como un intermediario de software que permite que dos aplicaciones se comuniquen entre sí [19].

puedan provocar. En particular, desde un punto de vista interno, uno de los focos de atención más importantes debe ser el cuidado de la plantilla de trabajadores y sus condiciones laborales. Para ello, es importante conocer su estado y las opiniones que tengan sobre su puesto y el ambiente laboral.

Este tipo de información se puede obtener de una manera interna, cómo puede ser realizando reuniones internas o encuestas para conocer la situación del trabajador cada cierto tiempo. Pero también existen fuentes de datos externas que posibilitan la obtención de esa información. En las páginas web de empleos existen ciertos apartados sobre las valoraciones que han tenido los propios trabajadores sobre las empresas, y esta información puede ser extraída para hacer uso de ella internamente y poder tomar decisiones que beneficien a los intereses de los empleados y así poder mejorar la imagen interna y externa de la empresa.

La principal motivación de este proyecto es el aprovechamiento de la gran cantidad de información que hay en la web y en concreto sobre valoraciones de empresas por parte de sus propios empleados. De este modo, se puede hacer uso de esta información aplicando técnicas de procesamiento del lenguaje natural para la clasificación de la información.

Esto permitirá conocer analizar el sentimiento detrás de las valoraciones de la empresa que hacen sus propios empleados, pudiendo ser esto un paso previo a la identificación de los problemas principales mencionados en las opiniones. Haciendo uso de las conclusiones obtenidas de estos análisis las empresas podrían tomar decisiones más eficaces y precisas para mejorar la vida laboral de sus empleados.

1.2 DESCRIPCIÓN DE LAS TECNOLOGÍAS

En el siguiente apartado, se detallan las tecnologías que han sido necesarias para la realización del proyecto.

1.2.1 LENGUAJE DE PROGRAMACIÓN

El lenguaje de programación utilizado para el desarrollo del proyecto ha sido **Python**. Se trata de un lenguaje de programación interpretado, orientado a objetos y de alto nivel con semántica dinámica. Posee una estructura de alto nivel, lo que permite su rápido aprendizaje y sencillez en la programación. Esto hace que sea muy atractivo para el desarrollo de aplicaciones, así como para el desarrollo de scripts o como lenguaje intermedio para la integración de diferentes sistemas. Uno de los motivos principales por los que ha sido elegido para desarrollar el proyecto es que admite módulos y paquetes, lo que ha permitido disponer de todas las librerías necesarias para realizar el proyecto. En los siguientes apartados se detallan las librerías que han sido utilizadas para el desarrollo del proyecto, entre las que destacan BeautifulSoup [8] y Selenium [27] para el módulo de *web scraping*, y NLTK [22] para el módulo de NLP.

1.2.2 ENTORNO DE DESARROLLO INTEGRADO (IDE)

El entorno de desarrollo integrado utilizado para la realización del proyecto ha sido Visual Studio Code (VSC) [29]. Este entorno de desarrollo integrado consigue, con la extensión Microsoft Python, ser un excelente editor de Python. Funciona en cualquier sistema operativo, proporciona autocompletado, depuración y pruebas unitarias. Pero principalmente ha sido elegido por su integración y fácil intercambio entre entornos virtuales. También permite el control de los cambios con su integración con Git.

1.2.3 CONTROL DE VERSIONES

El software elegido para el control de versiones ha sido Git [11]. Consiste en un sistema de control de versiones distribuido y gratuito, diseñado para realizar todo tipo de proyectos de desarrollo de software. Está ampliamente extendido como software de control de versiones lo que permite su sencilla incorporación al proyecto.

En cuanto al repositorio dónde se ha almacenado todo el código desarrollado y controlado mediante Git, se ha utilizado BitBucket. Un repositorio de código versionado con Git es una

herramienta de colaboración y alojamiento de desarrollo software. Permite a los desarrolladores trabajar de forma paralela mediante ramas de trabajo individual, que posteriormente se incorporan en una rama general dónde se encuentra todo el código del proyecto.

Capítulo 2. DEFINICIÓN DEL TRABAJO

2.1 OBJETIVOS

El objetivo principal del proyecto es el **análisis de viabilidad de la aplicación de técnicas de procesamiento del lenguaje natural para el análisis de sentimiento en valoraciones**.

Para el cumplimiento del objetivo principal del proyecto se han definido una serie de objetivos específicos:

- Exploración e implementación de un módulo de *web scraping*.
- Desarrollo de modelos de clasificación mediante el uso de técnicas del procesamiento del lenguaje natural.
- Exploración de las aplicaciones de metodologías NLP en contextos realistas.

2.1.1 EXPLORACIÓN E IMPLEMENTACIÓN DE UN MÓDULO DE *WEB SCRAPING*

El objetivo específico se ha definido para la extracción de las valoraciones que tienen las empresas en una página web de empleo. Las valoraciones han sido realizadas por trabajadores de la empresa o antiguos trabajadores. Esta información extraída permite a cualquier a empresa tener una información muy valiosa sobre su imagen.

2.1.2 DESARROLLO DE MODELOS DE CLASIFICACIÓN MEDIANTE EL USO DE TÉCNICAS DEL PROCESAMIENTO DEL LENGUAJE NATURAL.

El siguiente objetivo específico consiste en el desarrollo de modelos utilizando técnicas de NLP para la correcta clasificación de las valoraciones. Este desarrollo permitirá solucionar un problema en su versión más elemental, y que pueda ser extendido hacia contextos más realistas.

2.1.3 EXPLORACIÓN DE LAS APLICACIONES DE METODOLOGÍAS NLP EN CONTEXTOS REALISTAS

Este objetivo secundario es un objetivo que se nutre de la implementación y desarrollo de los anteriores objetivos específicos. Consiste en el análisis de la implementación de la solución a otros contextos realistas. A lo largo del trabajo se mencionan algunas de las posibles vías de desarrollo en otros contextos.

2.2 METODOLOGÍA

En cuanto a la metodología utilizada para el desarrollo del proyecto se ha separado en dos módulos de desarrollo. El primer módulo se corresponde con el desarrollo del módulo *web scraping* para la extracción de los datos que se van a utilizar para el resto del proyecto. El segundo módulo se corresponde con aplicación de técnicas de procesamiento del lenguaje natural que aportará el valor y el análisis de sentimiento de las valoraciones.

2.2.1 MÓDULO WEB SCRAPING

A continuación, se exponen los pasos utilizados para el desarrollo del módulo de *web scraping*. En la siguiente lista de preguntas se diferencian los pasos a seguir para implementar un módulo de *web scraping*:

1. ¿Tienen una **API**? ¿Ofrece los datos que busco?
2. ¿Es **legal** scrapear la información que necesito?
3. ¿Puedo **interceptar** peticiones?
4. ¿Puedo **replicar** esas peticiones con Request y BeautifulSoup?
5. Uso de Chromedriver y **Selenium**.

Otro de los aspectos a tener en cuenta para el desarrollo de un módulo de *web scraping*, son los bloques principales en los que se dividen las diferentes tareas a realizar. La gran mayoría

de los procesos de *web scraping* se pueden dividir en los tres bloques de desarrollo mostrados en la Figura 1.

Figura 1. Procesos de proyecto web scraping

El principal motivo de la separación de dichos procesos es la posibilidad de su ejecución en diferentes momentos. A continuación, se detalla en qué consiste cada uno:

- ***Spider***: proceso inicial que consiste en extraer todo lo que contiene información útil y guardar su referencia para la fase de extracción (*scraper*). Este proceso puede ejecutarse diariamente para comprobar si se tiene nueva información útil, actualizar la información general, etc.
- ***Scraper***: este proceso es el principal y se aprovecha de la información que se ha obtenido en la fase de *spider*. Utiliza las referencias (URLs[2], número de valoraciones, etc.) para realizar el proceso de extracción de la información útil requerida. Este proceso es importante al principio para extraer toda la información, y no es ejecutado nuevamente hasta que se disponga de información nueva.
- ***Parser***: es el proceso final que se ejecuta para limpiar y procesar toda la información obtenida en el proceso anterior y que se pueda empezar a utilizar en la parte del proyecto correspondiente a la ciencia de datos (en este caso NLP).

[2] Es el acrónimo de *Uniform Resource Locator*, es la dirección de un determinado recurso único web.

Una correcta separación de estos procesos permite evitar la concurrencia de actividades exigentes en términos de capacidad de computación, de modo que la actualización de información pueda ser llevada a cabo sin mucho coste computacional.

2.2.2 METODOLOGÍA NLP

Este apartado describe la metodología seguida en un proyecto de NLP de estas características.

Como se explica en el capítulo introductorio, el procesamiento del lenguaje natural es una técnica de procesamiento que permite a las máquinas interpretar el lenguaje humano. Dentro del mundo de los datos, el lenguaje (tanto humano como escrito) se clasifica dentro de los datos desestructurados. En el presente proyecto, se va a estudiar el lenguaje humano, en concreto valoraciones sobre una empresa. Estas valoraciones se van a poder clasificar y procesar en función diferentes aspectos (puesto del trabajador, localización, puntuación, etc.).

Los principales análisis que se pueden desarrollar en un proyecto de NLP son los siguientes:

- **Análisis léxico**: consiste en la división de un texto en la principal estructura jerárquica de cualquier texto. Esta estructura es: texto, párrafos, frases y palabras. Este análisis sirve a modo de exploración inicial y entra dentro de la primera fase del proyecto.
- **Análisis sintáctico**: este tipo de análisis comienza a tratar las palabras de cada texto y realizar ciertas transformaciones para obtener información útil de ellas. Es un paso importante de cara a los siguientes análisis en una metodología NLP. Entre los tipos de análisis sintácticos se encuentran:
    - *Lemmatization*
    - *Stemming*
    - Segmentación morfológica
    - Etiquetado POS

  - Segmentación de las palabras
- **Análisis semántico**: este análisis se centra en extraer significado del texto y analizar el sentido de la frase. Se encuentran los siguientes tipos de análisis:
  - Clasificación de tópicos
  - Análisis de sentimientos
  - Clasificación de intenciones
  - NER
  - WSD
  - Co-referencias
- **Análisis del discurso**: análisis más profundo que tiene en cuenta el contexto del texto. No forma parte del alcance del proyecto.
- **Análisis pragmático**: análisis centrado en la comunicación e interpretación global del lenguaje. No forma parte del alcance del proyecto.

En cuanto a la metodología seguida para el desarrollo de esta parte del proyecto ésta consta de las siguientes partes:

- **Preprocesado del texto**: paso fundamental dentro de un proyecto de NLP- En función del objetivo del análisis se pueden realizar diferentes tipos de preprocesados. En este proyecto se ha aplicado un preprocesamiento mencionado anteriormente en el análisis sintáctico, el *stemming* que consiste en la búsqueda de la raíz mediante la reducción de las palabras a su raíz o base.
- **Extracción de *features***: transformación de la información desestructurada en variables aprovechables por parte de los modelos de clasificación para el análisis de sentimientos posterior.
- **Desarrollo del modelo**: construcción de los modelos de clasificación para la extracción de la información relevante contenida.
- **Entrenamiento y test del modelo**: la información extraída y procesada es introducida en los diferentes modelos de clasificación para la obtención de

información relevante contenida y la validación de los modelos con el conjunto de test.

- **Análisis de los errores**: identificación de los patrones en el texto que los modelos no han sido capaces de identificar correctamente. Esto permite a través de un proceso iterativo lograr un mejor ajuste de los modelos.

2.3 ESTRUCTURA DEL DOCUMENTO

En la memoria se explican los conceptos básicos para la familiarización con el *web scraping* y el procesamiento del lenguaje natural, así como la descripción del trabajo realizado para el logro de los objetivos. Está dividida en las siguientes secciones:

- En el Capítulo 1. se presenta el contexto de los módulos de *web scraping* y de NLP, se describe la motivación principal que ha impulsado al desarrollo del proyecto, y se explican las principales tecnologías utilizadas.
- En el Capítulo 2. se realiza la descripción del trabajo. La descripción incluye los objetivos principales y secundarios del proyecto, realiza una breve explicación de las metodologías de trabajo que se han seguido y termina con la estructura de la memoria realizada.
- En el Capítulo 3. se realiza una descripción de los conceptos fundamentales de las técnicas de *web scraping*. Estos conceptos serán utilizados para la descripción del trabajo desarrollado y son fundamentales para el entendimiento de módulo desarrollado.
- En el Capítulo 4. se realiza una descripción de los fundamentos de los modelos de procesamiento del lenguaje natural. También incluye una breve explicación de las aplicaciones que el procesamiento del lenguaje natural puede tener, y las técnicas utilizadas en el proyecto.

- En el Capítulo 5. se describe el problema en términos de datos de entrada, requisitos del modelo, datos de salida y se realiza un repaso del estado del arte de los modelos de procesamiento del lenguaje natural.
- En el Capítulo 6. se describen cada uno de los pasos realizados para la implementación del módulo de *web scraping*.
- En el Capítulo 7. se explica la formulación de los modelos implementados. También se muestran detalles de la construcción de los modelos en la interfaz de desarrollo.
- En el Capítulo 8. se describen los resultados obtenidos y la información relativa a la explotación de los resultados.
- En el Capítulo 9. se explican las conclusiones obtenidas a lo largo de la ejecución del trabajo, describiendo la evaluación del cumplimiento de los objetivos. También recoge las ideas surgidas durante el desarrollo del trabajo, de forma que queden registradas para trabajos futuros.

Capítulo 3. WEB SCRAPING

El origen del *web scraping* se remonta a la época en la que nació la *World Wide Web* (WWW) en 1989, y fue en junio de 1993 cuando se creó el primer robot web. El robot se llamaba World Wide Wanderer [30] y su principal objetivo era únicamente la medición del tamaño de la web. En diciembre de 1993, se lanzó el primero robot de búsqueda web basado en un rastreador, cuyo funcionamiento consistía en indexar enlaces (recopilar y editar los enlaces en un formato determinado) y ofrecía una búsqueda lineal sin clasificar los resultados.

Más adelante, concretamente en el año 2000 nació la primer API Web, facilitando el desarrollo de los programas y el desarrollo Web. Y también en el año 2000 se lanzaron dos APIs que permitían a los programadores acceder y descargar algunos de los datos disponibles al público de esa API. Ese fue un momento muy importante, ya que desde entonces muchos sitios web ofrecen su API para que se pueda acceder a su base de datos pública.

Con el aumento de la cantidad de información disponible en la WWW en los últimos años, se ha popularizado una técnica que es capaz de obtener mucha de la información de la que se dispone. Esta técnica se llama *web scraping* y consiste en la recopilación y extracción de la información disponible en la WWW de una forma automatizada Existen multitud de posibilidades para la extracción de la información, y en muchas ocasiones no es una tarea sencilla, ya que hay muchas páginas web que intentan impedir que el acceso a la información de su web sea sencillo.

En los siguientes apartados se detallan los conocimientos teóricos para comprender cómo se realizan este tipo de técnicas de extracción de datos, las posibles aplicaciones que tienen y las tecnologías utilizadas para el desarrollo del módulo de *web scraping* del proyecto.

3.1 DESCRIPCIÓN TEÓRICA

Este apartado reúne una introducción a todas las tecnologías que influyen en la realización de un proyecto de *web scraping*. El orden de los apartados a explicar tiene relación con los pasos a seguir dentro de la metodología de un módulo *web scraping* explicada en el punto 2.2.1.

3.1.1 MODELO CLIENTE – SERVIDOR

La arquitectura cliente – servidor es un modelo informático que se basa en que el servidor aloja y gestiona todos los recursos y servicios disponibles que va a consumir el cliente. Este tipo de arquitectura es también conocida como modelo de computación en red o red cliente - servidor, ya que todas las peticiones se realizan y se entregan a través de la red. En la Figura 2 se muestra un ejemplo de arquitectura cliente – servidor compleja:

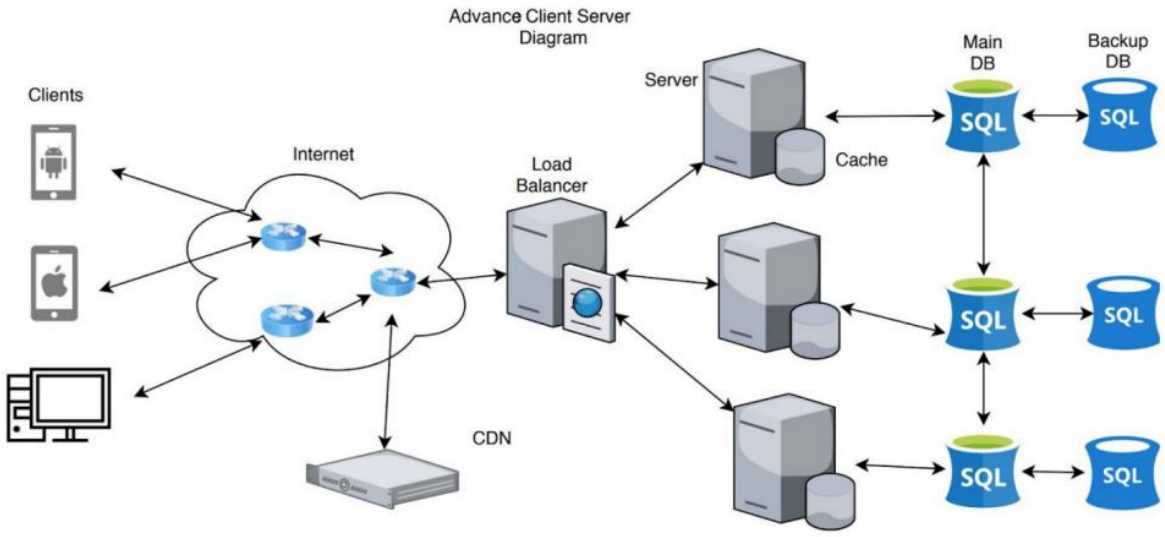


Figura 2. Modelo cliente – servidor [21].

En ella se muestra cómo los clientes se conectan a través de la red de Internet a los servidores. CDN se corresponde con la red de distribución de contenido que permite reducir los tiempos

de carga de contenidos de páginas web al disminuir la distancia física entre el servidor y el cliente. Por último, el balanceador de carga distribuye las peticiones en función de la carga que tenga cada servidor.

El proceso de una petición sería el siguiente:

1. **Petición del cliente al servidor**: el cliente desde su dispositivo informático realiza una petición de información al servidor (Ejemplo: un usuario accede a la página web de Indeed y pincha en la pestaña de valoraciones de empresa.).
2. **El servidor asignado recibe la petición**: el servidor que ha recibido valida que la petición se ha realizado correctamente y obtiene la información de la base de datos para devolvérsela al cliente.
3. **El servidor envía la respuesta al cliente**: el cliente recibe la respuesta del servidor (Ejemplo: el usuario accede a la pestaña de valoraciones y se cargan en el navegador.).

3.1.2 APIs

API es el acrónimo de *Application Programming Interface* (Interfaz de Programación de Aplicaciones), que consiste en ser un intermediario de software que permite que dos sistemas software se comuniquen entre sí.

Las APIs suelen estar formadas por diferentes partes que están a disposición del programador y que conforman los extremos del canal de comunicación que se ha mencionado en la definición. A estos extremos de la comunicación se les llama *endpoints* o puntos finales. Los puntos finales pueden incluir una URL de un servidor y proporcionan el punto de acceso desde el que la API se conecta con el servidor y accede a los recursos que se han solicitado a través de la petición. Las APIs funcionan de una forma similar a lo explicado en el modelo cliente – servidor a través de peticiones y respuestas. Las APIs realizan peticiones a los puntos finales y estas reciben una respuesta. Una API bien implementada debe tener una

correcta documentación para que el programador sea capaz de realizar las peticiones al servidor y obtener la información que se requiera.

Por otro lado, uno de los propósitos de las APIs es ocultar los detalles internos del funcionamiento de un sistema, proporcionando una capa de seguridad y exponiendo sólo aquellas partes que un programador encontrará útiles, aunque los detalles internos cambien posteriormente. Por ejemplo, los datos del teléfono de un usuario nunca están totalmente expuestos al servidor y, del mismo modo, el servidor nunca está totalmente expuesto al teléfono del usuario.

El tipo de API más extendido y utilizado es la **API REST** o también conocida como API RESTful, y se trata de una interfaz de programación de aplicaciones que se ajusta a las restricciones del estilo de arquitectura software REST (Transferencia de estado representativa) y permite la interacción con servicios web RESTful.

REST [13] consiste en un estilo arquitectónico de software que define el conjunto de reglas para crear un servicio web. Los servicios web que siguen este estilo arquitectónico se llaman servicios web RESTful. Lo elementos más relevantes de un sistema RESTful son el cliente, que realiza una petición de recursos, y el servidor que dispone de esos recursos y se los envía al cliente (Modelo cliente - servidor). En un servicio web RESTful, las solicitudes realizadas a la URL de un recurso obtienen una respuesta con una carga útil formateada en HTML, XML, JSON o algún otro formato. Por ejemplo, la respuesta puede confirmar que el estado del recurso ha cambiado. La respuesta también puede incluir enlaces de hipertexto a recursos relacionados.

El protocolo de comunicación entre dos sistemas basados en REST es el protocolo **HTTP** [14], que es el acrónimo de Protocolo de Transferencia de Hipertexto de Internet. Proporciona operaciones (métodos HTTP) como GET, POST, PUT y DELETE. Estas operaciones permiten obtener información, publicar información, modificarla o eliminarla del servidor.

En la Figura 3 se muestra cómo sería la comunicación cliente servidor mediante el protocolo HTTP.

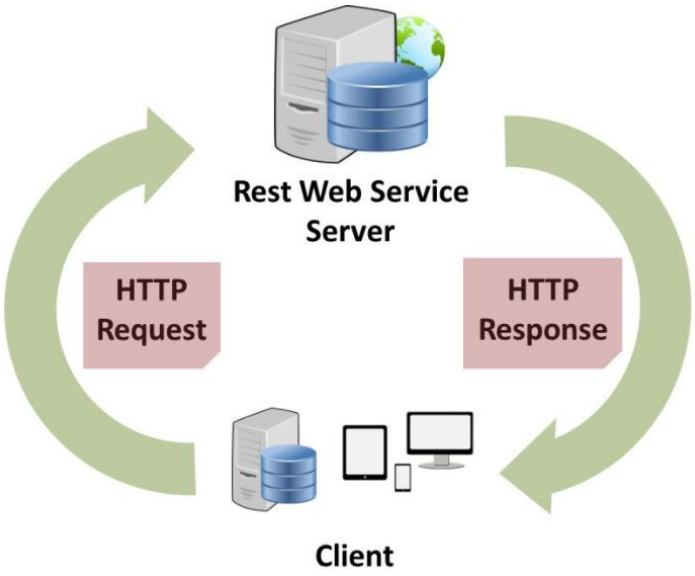


Figura 3. Ejemplo de comunicación REST [23].

Este tipo de comunicación permite la mayor interoperabilidad entre clientes y servidores en un entorno a escala de Internet de larga duración que traspasa las fronteras organizativas (de confianza).

3.1.3 HTML

Otro elemento fundamental relacionado con el *web scraping* es **HTML** (Lenguaje de Marcas de Hipertexto) que es un lenguaje informático sobre el que se desarrollan la mayoría de las páginas web y aplicaciones en línea. Un hipertexto es un texto que se emplea para referenciar a otros fragmentos de texto, mientras que un lenguaje de marcado de texto es una serie de marcas que indican a los servidores web el estilo y la estructura del documento.

HTML no se termina de considerar un lenguaje de programación, ya que no permite la creación de funcionalidades dinámicas. Lo único que permite es la creación y estructuración de secciones, párrafos y enlaces mediante etiquetas y atributos.

Los documentos HTML son archivos que terminan con la extensión .html o .htm. El navegador web lee este archivo y representa su contenido para que el usuario final pueda verlo. Los archivos HTML están formados por una serie de elementos, que no son más que los componentes básicos del lenguaje marcados con una etiqueta y uno o varios atributos. Las etiquetas informan al navegador dónde empieza y dónde acaba un componente, mientras que el atributo describe las características principales de un elemento (ancho, largo, color, etc.).

En la Figura 4 se pueden observar las tres partes principales de un elemento de HTML:

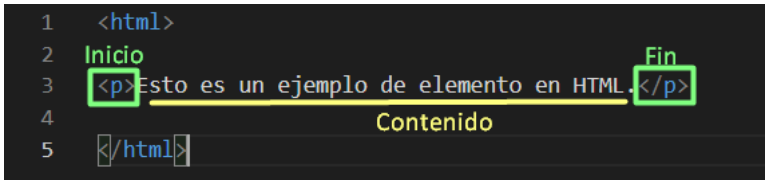


Figura 4. Elemento de HTML.

Se puede ver como todo elemento está formado por una etiqueta de inicio o creación del elemento, por el contenido del elemento (en este caso es un párrafo con un texto) y una etiqueta de fin del elemento.

La mayoría de las páginas web disponen de tres etiquetas que son las principales:

- <html>: es la raíz del archivo y define todo el documento como un documento de tipo HTML.
- <head>: Contiene la información principal de la página, como puede ser el título o atributo charset[3].
- <body>: es la etiqueta principal que contiene todo el contenido de la página web.

[3] Atributo que especifica la codificación de caracteres del documento HTML.

Existen otro tipo de elementos dentro de una página web entre los que destacan los enlaces (que están marcados con la etiqueta <a>) y los botones (marcados con la etiqueta <button>) que son importantes dentro de un módulo de *web scraping*.

3.2 APLICACIONES DEL WEB SCRAPING

En este apartado se detallan los principales tipos de *Web scraping* que existen, sus aplicaciones prácticas y los impedimentos que se pueden encontrar en cada una de ellas.

Existen diferentes técnicas para hacer *Web scraping*. En la siguiente lista se muestran las más conocidas:

- **Copy-pasting**: esta técnica se corresponde con el scraping manual, y lo que se hace es copiar y pegar el contenido de la Web. Esta forma no se utiliza ya que conlleva mucho tiempo y es un proceso muy repetitivo, por lo que se suelen buscar otras alternativas para realizar el *Web scraping*.
- **HTML parsing**: esta técnica consiste en el análisis sintáctico del archivo HTML. Se suele realizar con JavaScript y se dirige a páginas web lineales o anidadas que se caracterizan por tener una estructura subyacente similar que el programa de *Web scraping* puede detectar y recoger la información que le interese.
- **DOM parsing**: el DOM define la estructura del documento y el contenido. Por lo que los *scrapers* hacen uso de esa estructura para extraer una visión de la forma general de la página web.
- **Agregación vertical**: esta técnica es usada por las empresas que tienen acceso a una gran potencia informática. Son técnicas que no requieren intervención humana y que crean y supervisan bots que realizan todo el proceso de extracción de datos. La calidad de esta técnica es medida en función de la calidad de los datos que extraen y de la cantidad de información que recogen.
- **Coincidencia de patrones de texto**: es una técnica que implica el uso del comando grep de UNIX, que realiza un rastreo de la devolución de un comando de UNIX y si

coincide con el texto que se le indica devuelve la búsqueda. Existen algunas herramientas que se basan en este tipo de técnica.

- **Programación HTTP**: este tipo de técnicas se basan en el protocolo HTTP y consisten en realizar peticiones al servidor web remoto y en analizar la respuesta obtenida.
- **Selenium**: esta técnica consiste en hacer uso de Selenium, que es una herramienta de automatización del navegador web y que permite realizar muchas acciones simulando la interacción humana con el navegador. Esta técnica permite la obtención de datos precisos. Se trata de una herramienta ampliamente utilizada no sólo para el *web scraping*, ya que es utilizada por muchos desarrolladores para realizar pruebas de las páginas web implementadas.

Una vez enumeradas las principales técnicas de *web scraping* existentes, se analizan las posibles aplicaciones prácticas que tienen este tipo de técnicas.

A continuación, se exponen cuatro casos de uso en los que es necesario realizar *web scraping* para poder acceder a los datos valiosos de cada uno de ellos.

El primer caso de uso presenta cierta similitud con el trabajo desarrollado en el presente proyecto, pues tiene que ver con el análisis de las **redes sociales**. Aunque la mayoría de las redes sociales tienen una API para poder acceder a la información, en algunas situaciones no es suficiente. Estas casuísticas pueden ser: que la API pública esté limitada a una serie de peticiones y que se necesite acceder a más información, o que las publicaciones son borradas y no se puede acceder más a ellas. Es por esto por lo que en muchas ocasiones se proceden a realizar módulo de *web scraping* que obtengan la información que se publica en las redes sociales ya sea en tiempo real o cada poco tiempo, para no perder nada de información. En cuanto a la utilidad de obtener este tipo de información, ésta presenta un valor cuando se analiza en conjunto, porque se pueden identificar patrones de conducta o los temas más repetidos. En el trabajo descrito en este documento se ha llevado a cabo el análisis de

sentimientos de forma similar a los que se suelen realizar en este tipo de proyectos de análisis de redes sociales.

El segundo caso de uso se relaciona con la **inversión**. En muchas ocasiones la información valiosa para los inversores inmobiliarios no está accesible directamente en el formato adecuado para tratarla y analizarla. Por lo tanto, si se quiere obtener en un formato adecuado para el tratamiento de la información, es necesario la implementación de un módulo de *web scraping*.

El tercer caso de uso pone de manifiesto los beneficios que aporta la unión con los modelos de ***Machine Learning*.** Este tipo de modelos necesitan de muchos datos para poder evolucionar y mejorar sus resultados. Con la realización de un módulo de *web scraping* se pueden conseguir muchos datos precisos que realimenten a los modelos y en un tiempo relativamente corto.

Por último, se mencionan las medidas que las páginas web están implementando para evitar la extracción de la información contenida a través de técnicas de *anti-scraping* como las descritas a continuación:

- **Direcciones IP**[4]: una de las técnicas más sencillas para identificar el *web scraping* es el rastreo de la dirección IP. Mediante el seguimiento de la IP se puede conocer si el usuario que está accediendo a la información es una persona o un robot. Normalmente el comportamiento de un robot que ha sido programado para extraer la información de la web realiza muchas peticiones en poco tiempo. Esto permite, a través del rastreo de IPs, observar un comportamiento anómalo y poder bloquear esa IP para que no acceda a la información de la web. La forma más frecuente de realizar la medición es medir el número y la frecuencia de las visitas por unidad de tiempo.

[4] Acrónimo de Internet Protocol, es el protocolo que define el conjunto de reglas para la comunicación por medio de Internet.

- ***Captcha***: otra de las técnicas más utilizadas es el *Captcha*, que consiste en un programa automático que determina si el usuario que hace uso de la página web es un humano o un robot. Existen diferentes tipos de pruebas que pueden consistir en identificar imágenes, letras o rellenar espacios en blanco de forma que sólo puedan ser resueltas por un humano. Este tipo de pruebas están en constante evolución, pero también han evolucionado las formas de poder solucionar el *Captcha* y seguir haciendo *web scraping*.
- **Registrarse - iniciar sesión**: otra técnica que utilizan muchas redes sociales es la identificación del usuario, por medio del inicio de sesión, para poder acceder a la información de la página web. Esto implica que el módulo de *web scraping* debe iniciar sesión, almacenar las cookies de la sesión e introducirlas en sesiones posteriores para lograr acceder a la información.
- **Agente de usuario**: el agente de usuario se encuentra dentro de la cabecera de una petición HTTP y sirve para identificar la forma en la que el usuario está visitando la web. En concreto el agente de usuario almacena el sistema operativo, el navegador, el idioma del navegador, etc. Esto permite al sistema *anti-scraping* identificar a los robots en caso de que no se envíe un agente de usuario en la cabecera. El módulo de *web scraping* por lo tanto debe realizar peticiones como si fuese un navegador con agente de usuario en la cabecera.
- **AJAX**: este tipo de técnicas obligan al módulo de *web scraping* a navegar a través de la página web con el script, ya que no se puede acceder a la información valiosa hasta que se ha ido navegando a través de ella haciendo clic en algunos botones o teniendo que rellenar algún formulario. La URL permanece idéntica, aunque la página web esté cambiando debido a la interacción del usuario con la web.

3.3 TÉCNICAS Y RECURSOS EMPLEADOS

Durante el desarrollo del módulo *web scraping* se estudiaron diferentes formas de acceder a la información, y finalmente se han desarrollado dos formas distintas de poder extraer la información de la página web de empleo.

En el apartado 6.1 se explican las diferentes páginas web y las decisiones tomadas para el uso de las técnicas de *web scraping* que finalmente se han utilizado en el módulo.

Las técnicas empleadas para el desarrollo del módulo de *web scraping* han sido:

- **Programación HTTP**: para el uso de este tipo de técnica se han utilizado las librerías de Python que se llaman Requests y BeautifulSoup.
- **Selenium**: para el uso de esta técnica se ha utilizado la librería de Python llamada Selenium.

3.3.1 LIBRERÍA REQUESTS

Esta librería permite enviar peticiones HTTP de manera sencilla. En la Figura 5 se muestra un ejemplo sencillo de una petición HTTP realizada con la librería requests de Python.

```
1   import requests
2
3   url = "https://es.indeed.com/cmp/Burger-King/reviews"
4
5   payload={}
6   headers = {}
7
8   response = requests.request("GET", url, headers=headers, data=payload)
9
10  print(response.text)
11
```

Figura 5. Ejemplo de petición HTTP con Requests.

La respuesta se guarda en la variable "response" y en "response.text" se obtendría todo el código HTML de la página web a la que se ha lanzado la petición HTTP.

3.3.2 LIBRERÍA BEAUTIFULSOUP

Esta librería permite la extracción de datos de archivos HTML y XML. La función principal de la librería es analizar la estructura del archivo y la creación de un árbol de contenido que permite al usuario encontrar la información que necesita de una forma mucho más rápida y sencilla.

En la Figura 6 se muestra un ejemplo de cómo se realiza la extracción de la información a partir de la respuesta que se ha obtenido con la petición requests en formato HTML.

```
1   import requests
2   from bs4 import BeautifulSoup
3   url = "https://es.indeed.com/cmp/Burger-King/reviews"
4
5   payload={}
6   headers = {}
7
8   response = requests.request("GET", url, headers=headers, data=payload)
9
10  soup = BeautifulSoup(response.text, 'html.parser')
11
12  print(soup.find('a'))
```

Figura 6. Ejemplo de BeautifulSoup con HTML.

El script realiza una petición HTTP al servidor web de la página Indeed y su respuesta es enviada como entrada a la función de BeautifulSoup para extraer el árbol de contenido y acceder a la información de una manera más rápida. Finalmente imprime por pantalla el primer enlace (etiqueta <a>) que se encuentre en el archivo HTML.

3.3.3 LIBRERÍA SELENIUM

La librería Selenium está diseñada para ser una herramienta de testeo de las páginas web, pero al permitir el control de un navegador web, posibilita la utilidad de esta librería para la implementación de un módulo de *web scraping* que extraiga información de las páginas web.

Esta librería permite el control de distintos navegadores web. En el presente trabajo se ha utilizado la librería Selenium para controlar el navegador de Chrome. Esta librería permite extraer información de la página web a la vez que se navega. En la Figura 7, se muestra un ejemplo de script que utiliza Selenium. El código abre un navegador de Chrome, se redirige a la URL definida en el script, e imprime por pantalla el primer enlace (etiqueta <a>) que se encuentra en el archivo HTML de la página web controlada por Selenium.

```
import traceback
# Time
import time
# Web Scrapping tools
from selenium import webdriver

url = "https://es.indeed.com/?from=gnav-acme--discovery-webapp"

try:
    # Chromedriver
    chromeOptions = webdriver.chrome.options.Options()
    chromeOptions.add_argument("--start-maximized")
    driver = webdriver.Chrome(executable_path="chromedriver.exe", chrome_options=chromeOptions)

    driver.get(url)
    time.sleep(5)
    print(driver.find_element_by_tag_name('a'))

    driver.close()

except Exception as e:
    driver.close()
    print(traceback.format_exc())
```

Figura 7. Ejemplo con Selenium.

Capítulo 4. PROCESAMIENTO DEL LENGUAJE NATURAL

Los primeros enfoques conceptuales sobre el término de "máquina traductora" fueron a mediados de los años 30. A partir de la nueva tecnología surgieron dos patentes, la primera fue registrada por Georges Artsrouni y utilizaba un diccionario bilingüe para traducir las palabras de un idioma a otro utilizando cinta de papel. La segunda patente pertenecía a Peter Troyanskii [25] que trazó una estrategia para abordar la gramática de una lengua y poder traducir de un idioma a otro en función de las reglas gramaticales generales.

Sin embargo, no fue hasta los años 40 cuando se realizó el primer intento de realizar una traducción automática de idiomas. Surgió durante la Segunda Guerra Mundial como intento de descifrar los códigos cifrados para enviar mensajes en la Guerra.

Desde entonces ha habido ciertos acontecimientos que han supuesto un paso hacia adelante en el mundo del procesamiento del lenguaje natural. En la Figura 8 se muestra un resumen de los más importantes:

Figura 8. Evolución del NLP [6].

A continuación, se enumeran y describen brevemente dichos desarrollos:

- ***Bag of words***: es la versión más simple de representar un texto en forma numérica, para que un ordenador sea capaz de interpretarlo y entenderlo. Consiste en realizar un diccionario de palabras que haga un recuento de cada palabra para tener un control del número de veces que aparece para término dentro del texto.

- **TF-IDF**: es el acrónimo de frecuencia de términos – frecuencia inversa de documentos. Consiste en una estadística numérica que intenta reflejar la importancia de que un término aparezca dentro de un documento que forma parte de una colección de documentos.
- **Matriz de coocurrencias**: esta técnica sirve para tener un registro del número de veces que aparecen dos palabras en un mismo contexto. Lleva un recuento del número de veces que dos palabras aparecen en la misma frase. Esta matriz es fundamental en la siguiente técnica.
- **Word2Vec**: esta técnica permite extraer el contexto de una palabra dentro de un texto. Se define como la representación vectorial de una palabra.
- ***Transformer Models***: este tipo de técnica destaca por identificar las relaciones entre las frases de un mismo documento. A modo explicativo a continuación se muestra un ejemplo sencillo: "Juan realizó un estupendo entrenamiento. El entrenador le felicitó, pero él tiene que seguir progresando". En la frase el *transformer* es capaz de identificar que el pronombre "él" se refiere a Juan. Esto permite al modelo tener una comprensión lectora que no existía en técnicas anteriores.
- **XLNet**: Son técnicas de alta complejidad utilizadas para labores como la inferencia del lenguaje natural, la respuesta a preguntas, el análisis de sentimientos y la clasificación de documentos.

4.1 INTRODUCCIÓN

Este apartado reúne todos los conocimientos teóricos necesarios sobre las técnicas que se han empleado en el desarrollo del proyecto, así como otras técnicas que se han estudiado para desarrollos futuros.

4.1.1 REGRESIÓN LOGÍSTICA

La regresión logística modela las probabilidades de un problema de clasificación con dos posibles salidas [17]. Es una extensión de la regresión lineal para un problema de clasificación.

Funciona de tal forma que modela la probabilidad de que un registro tome uno de los dos valores. El modelo no realiza la clasificación, la clasificación se realiza a partir de la probabilidad devuelta por el modelo seleccionando un umbral de decisión entre ambas clases.

Este tipo de modelos es idóneo cuando se pretende obtener un modelo que sea interpretable, ya que permite entender el impacto de cada predictor sobre la variable de salida.

Para devolver una probabilidad entre 0 y 1, el modelo de regresión logística hace uso de la función Sigmoide. La función Sigmoide se representa en la Figura 9:

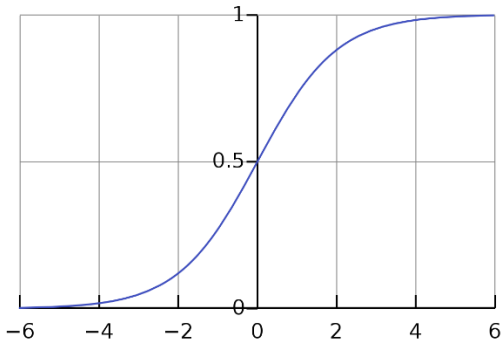


Figura 9. Función Sigmoide [31].

Esta función se define en la siguiente fórmula:

$$S(x) = \frac{1}{1 + e^{-x}}$$

Dónde para valores altos de la *x* la salida de función tiende a ser 1, y para valores bajos de la *x* la salida tiende a ser 0.

4.1.2 NAIVE BAYES

El modelo de clasificación de Naive Bayes es un modelo de *Machine Learning* probabilístico que se utiliza para resolver problemas de clasificación. Está basado en el teorema de Naive Bayes, que hace uso de la probabilidad condicionada (probabilidad de que ocurra un evento A sabiendo que ha ocurrido un evento B) [26]. El teorema de Naive Bayes es el siguiente:

$$P(A|B) = \frac{P(B|A)\,P(A)}{P(B)}$$

Esta fórmula se traslada al problema de clasificación de la siguiente manera:

- *P(B|A)*: la probabilidad de que sea B, habiendo ocurrido A, se corresponde con el parámetro de verosimilitud (*likelihood*).
- *P(A)*: la probabilidad de A se corresponde con la probabilidad a priori de que un registro aleatorio sea de la clase A.
- *P(B)*: la probabilidad de B se corresponde con la probabilidad del predictor, es decir, la probabilidad de que un registro aleatorio contenga el predictor B.

El nombre del algoritmo *Naive* significa ingenuo y proviene de la suposición de que las variables de entrada son mutuamente independientes.

4.1.3 LIBRERÍAS PRE-ENTRENADAS

Además de la implementación completa de los dos métodos descritos anteriormente, en el proyecto se han utilizado dos librerías pre-entrenadas para realizar un análisis del sentimiento de las valoraciones extraídas en el módulo *web scraping*. En este apartado se realiza una descripción teórica de las librerías utilizadas.

4.1.3.1 NLTK VADER

VADER (*Valence Aware Dictionary for Sentiment Reasoning*) es un modelo de la librería NLTK que se utiliza para el análisis de sentimientos. Permite la interpretación de los sentimientos en la clasificación positivo/negativo, como en la intensidad de la emoción.

Está basado en un diccionario (*Bag of words*) que asigna una puntuación de sentimiento a cada palabra recogida en el texto, de tal forma que la puntuación total obtenida se corresponde con la nota de sentimiento que el modelo devuelve a ese texto. Además, es un modelo basado en reglas, es decir, utiliza una serie de reglas gramaticales y convenciones sintácticas que le permiten expresar la polaridad e intensidad del sentimiento.

VADER utiliza el léxico y las reglas gramaticales para poder analizar el sentimiento de cada frase. La utilidad de las reglas gramaticales aplica principalmente en el análisis de las estructuras del texto, que permiten cambiar la polaridad de una palabra que en principio es positiva en una negativa, dependiendo del sentido que se haya utilizado de la palabra (Ejemplo: "muy bueno" no es lo mismo que "no muy bueno").

4.1.3.2 TextBlob

TextBlob es una librería de Python que se utiliza para el procesamiento del lenguaje natural, y que hace uso de la librería NLTK para realizar el preprocesado de los textos.

Esta librería también permite extraer tanto la polaridad como la subjetividad de una frase. Hace uso de un diccionario léxico que devuelve una puntuación a la palabra del texto y luego aplica reglas gramaticales que pueden invertir la polaridad de la palabra. Además, incluye etiquetas semánticas que permiten la interpretación de emoticonos, signos de exclamación, etc.

La puntuación de subjetividad que devuelve el modelo indica el grado de opiniones personales que contiene el texto. Para medir la subjetividad, analiza la intensidad que contiene el texto, que se define como la capacidad que tiene una palabra de modificar a la siguiente (Ejemplo: "**muy** malo" la palabra "muy" intensifica a la palabra "malo".).

4.2 APLICACIONES DEL NLP

El procesamiento del lenguaje natural tiene multitud de aplicaciones entre las que destacan: modelos de clasificación de texto, traducciones entre idiomas o auto correctores de texto

entre otros. Mezclados con técnicas más avanzadas como el Deep Learning se pueden realizar modelos de reconocimiento de audio para su transformación a texto. El proyecto recogido en el presente documento se ha centrado en modelos de clasificación de texto extrayendo el sentimiento, pero existen otras aplicaciones de clasificación de texto. A continuación, se mencionan algunas de las aplicaciones más conocidas en clasificación de texto:

- **Detección de spam**: clasificación de los correos como basura (peligrosos o basura) o un correo normal.
- **Monitorización de las redes sociales**: clasificación de los mensajes que se están monitorizando en las redes sociales, para la detección de mensajes que puedan tener un alto impacto para la intervención inmediata.
- **Clasificación de idiomas**: clasificación del idioma del texto. Este tipo de aplicación puede ser útil para la respuesta automática en un sistema de atención al cliente.
- **Automatización de procesos de atención al cliente**: sistema que detecta la urgencia de una petición y redirige hacia el centro de atención al cliente oportuno.
- **Identificación de autor**: modelo entrenado con textos escritos por un autor, o conjunto de autores, que detecta si un texto introducido al modelo ha sido escrito por él o por ellos.

4.3 TÉCNICAS EMPLEADAS

A lo largo del presente proyecto se han empleado como técnicas de clasificación de texto modelos de regresión logística, Naive Bayes y los modelos de las librerías pre-entrenadas descritos en el apartado 4.1.3.

A continuación, se detalla cada una de las técnicas empleadas para el desarrollo del proyecto.

4.3.1 PROCESAMIENTO REALIZADO

Antes de trabajar en la implementación de los modelos, es necesario realizar una serie de operaciones en los datos. Un conjunto de datos limpios permitirá a los modelos aprender las características relevantes del texto y evitar el aprendizaje del ruido de los datos.

En el presente proyecto se han realizado algunas operaciones sobre las valoraciones y se han aplicado algunas técnicas conocidas de NLP. Las técnicas que se han empleado para la limpieza de las valoraciones extraídas de Indeed han sido las siguientes:

4.3.1.1 Tokenization

La *tokenization* es el proceso que separa un texto en fragmentos de menos tamaño que se llaman tokens. El tamaño de los fragmentos puede variar (Frases, palabras, caracteres, etc.).

El tamaño de los fragmentos más utilizados es por palabras o por grupos de palabras (*n-grams*). Los grupos de palabras ayudan a los modelos a interpretar el contexto global de una frase.

4.3.1.2 Stemming

El *stemming* se define como el proceso de reducción de una palabra a su tallo. Esto se realiza mediante un proceso heurístico que corta los extremos de las palabras, logrando la supresión de los afijos derivativos. Forma parte del análisis sintáctico mencionado en el apartado 2.2.2 y es ampliamente utilizado por los motores de búsqueda para encontrar las mismas palabras o palabras similares a las de la búsqueda.

4.3.2 REGRESIÓN LOGÍSTICA

La regresión logística aplicada en problemas de clasificación hace uso de la función sigmoide mostrada en la Figura 9 para devolver una probabilidad de pertenecer a una clase u otra. La clasificación realizada en el proyecto es una clasificación binaria de las valoraciones que clasifica si una valoración ha sido positiva o negativa.

La regresión logística utiliza como parámetro de entrada una regresión lineal regular y aplica la función sigmoide para obtener una probabilidad entre 0 y 1.

La fórmula de la regresión lineal es la siguiente:

$$z = \theta_0 x_0 + \theta_1 x_1 + \ldots + \theta_N x_N$$

Siendo θ el vector de pesos de la regresión lineal. Este vector de pesos indica la importancia de cada una de las variables de entrada del vector x al modelo.

Por lo tanto, la fórmula de la regresión logística aplicada queda de la siguiente forma:

$$S(x^{(i)}, \theta) = \frac{1}{1 + e^{-\theta^T x^{(i)}}}$$

De tal forma que cuando $\theta^T x^{(i)}$ tenga un valor se acerque a ∞, el valor de salida de la regresión logística tienda a 1, si $\theta^T x^{(i)}$ se acerca a -∞, el valor de salida de la regresión logística tienda a ser 0.

En cuanto a la fase de entrenamiento del modelo para ajustar el vector de pesos θ de nuestro modelo de regresión logística, se ha hecho uso del algoritmo de optimización del descenso de gradiente junto con una función de coste que se explican en los siguientes apartados.

4.3.2.1 Función de coste

La función de coste del modelo de regresión logística implementado se define con la siguiente fórmula:

$$J(\theta) = -m1\sum i = 1m[y(i)\log S(x(i), \theta) + (1 - y(i))\log(1 - S(x(i), \theta))]$$

Se define como la media de la pérdida logarítmica en todo el conjunto de entrenamiento, donde:

- *m*: es el número de valoraciones del conjunto de entrenamiento.

- *y(i)*: es la etiqueta actual de la valoración del conjunto de entrenamiento.
- $S(z(\theta)^{(i)})$: es la predicción del modelo para la valoración actual del conjunto de entrenamiento.

La función de pérdida para una valoración del conjunto de entrenamiento se define como:

$$Loss = -1 \times [y^{(i)} \log S(x_{(i)}, \theta) + (1 - y^{(i)}) \log(1 - S(x_{(i)}, \theta))]$$

El valor negativo delante de la función de coste se debe a que todos los logaritmos van a ser negativos, ya que la función sigmoide *S* solo devuelve valores entre 0 y 1.

Esta función de coste tendrá un valor elevado en caso de la predicción realizada sea distinta de la etiqueta de la valoración.

4.3.2.2 Gradiente

Para el ajuste del vector de pesos θ del modelo, se hace uso del descenso de gradiente, que es un algoritmo de optimización iterativo que se utiliza para encontrar el valor mínimo de una función [24] (Peng Shi, 2020).

El gradiente de la función de coste con respecto a uno de los pesos del vector θ es:

$$\nabla_{\theta j} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} (S^{(i)} - y^{(i)}) x_j$$

Y para ajustar el peso θ_j del vector de pesos theta, se utiliza la siguiente fórmula:

$$\theta_j = \theta_j - \alpha \times \nabla_{\theta j} J(\theta)$$

Donde α es la tasa de aprendizaje con la que se quiere entrenar el modelo. Cuando menor sea la tasa de aprendizaje, la convergencia del modelo se alcanzará más lentamente.

El nivel de aprendizaje del modelo de regresión logística viene determinado por el descenso de gradiente de la función de coste definidos, la tasa de aprendizaje y el número de iteraciones que se realicen para que el entrenamiento termine.

4.3.3 NAIVE BAYES

El modelo de clasificación de Naive Bayes se basa en la aplicación del teorema de Bayes con hipótesis de independencia entre las variables de entrada. En término sencillos, un clasificador Naive Bayes asume que la presencia de una característica particular en una variable de entrada no tiene relación con ninguna otra característica de ninguna variable.

Para el entrenamiento del modelo es necesario realizar un cálculo de probabilidades previo. Para el ajuste del modelo se ha utilizado el entrenamiento de máxima verosimilitud, para el que no es necesario realizar una aproximación iterativa, como para el modelo de regresión logística explicado en el apartado anterior.

4.3.3.1 Cálculo de probabilidades

La primera probabilidad calculada es la probabilidad de que un registro pertenezca a cada una de las clases. En este proyecto en concreto se quiere desarrollar un modelo que clasifique las valoraciones entre buenas y malas. Para calcular la probabilidad de la clase se utiliza la siguiente fórmula:

$$P_{clase1}(V_{clase1}) = \frac{V_{clase1}}{V}$$

Dónde *V* es el número de valoraciones del conjunto de entrenamiento y V_{clase1} son el número de valoraciones de la clase 1.

La segunda probabilidad calculada es la probabilidad a priori. Se calcula de la siguiente manera:

$$logprior = \log\frac{P(V_{clase1})}{P(V_{clase0})} = \log\frac{V_{clase1}}{V_{clase0}} = \log(V_{clase1}) - \log(V_{clase0})$$

La tercera probabilidad calculada es la probabilidad positiva y negativa de una palabra. Para calcular esta probabilidad se hace uso del diccionario de frecuencias del conjunto de entrenamiento. El diccionario de frecuencias contiene el número de veces que una palabra ha aparecido en las valoraciones positivas y el número de veces que esa palabra ha aparecido en las valoraciones negativas. La fórmula para calcular esta probabilidad es la siguiente:

$$P(W_{clase1}) = \frac{frec_{clase1} + 1}{N_{clase1} + V}$$

Dónde N_{clase1} es el número total de las palabras que han aparecido en valoraciones de la clase1 y V es el total de palabras que se tienen en el vocabulario de las valoraciones (positivas y negativas). Nótese que en el numerador se suma uno a la frecuencia de que la palabra W haya aparecido en valoraciones de clase 1, para en caso de que no aparezca, esa palabra no tenga una probabilidad de 0.

4.3.3.2 Log Likelihood

Con el cálculo de las probabilidades anteriores, se puede calcular el valor de *log Likelihood* de cada palabra, haciendo uso de la siguiente función:

$$loglikelihood = \lambda\,(W) = \log\left(\frac{P(W_{clase1})}{P(W_{clase0})}\right)$$

Cuando se ha calculado el valor de *log Likelihood* de todas las palabras del conjunto de entrenamiento, se puede realizar la inferencia de una frase sumando el valor de *log Likelihood* de cada una de las palabras que aparecen en esa frase. Si el resultado es mayor que 0, se considera que esa valoración es positiva.

Capítulo 5. ESTADO DEL ARTE

En este apartado se lleva a cabo una revisión de la bibliografía existente sobre resoluciones de problemas similares. La revisión se ha realizado para los modelos de clasificación, excluyendo de este apartado la bibliografía de la implementación del módulo *web scraping* debido a que no suele ser objetivo de publicaciones científicas relevantes. Por lo tanto, la búsqueda se ha enfocado sobre modelos de clasificación de texto basados en el análisis de sentimientos, utilizando los modelos de regresión logística y de Naive Bayes. Además, se han investigado métodos que realizan la técnica conocida como *Ensemble* de modelos, que utilizan la salida de un conjunto de modelos para realizar la clasificación.

5.1 NAIVE BAYES Y REGRESIÓN LOGÍSTICA

A continuación, se resumen algunos *papers* científicos en los que se ha utilizado el modelo de clasificación de Naive Bayes:

En (Lopamudra Dey, 2016) se enuncia la aplicación de dos modelos de aprendizaje automático supervisado: Naive Bayes y K-Nearest Neighbour (k-NN). Los modelos realizan la clasificación de críticas de películas y de las valoraciones de hoteles, clasificando en valoraciones positivas o valoraciones negativas. Las conclusiones del trabajo desarrollado indican que el modelo de Naive Bayes ofrece mejores resultados en la clasificación de las críticas de películas y resultados similares al modelo de K-NN en las reseñas de los hoteles.

En (Hanhoon Kang, 2012) se describe la utilización del modelo de Naive Bayes para la mejora en la clasificación de valoraciones sobre restaurantes. El documento menciona la solución implementada ante un problema de clasificación desbalanceada, en el que los modelos desarrollados anteriormente obtenían mejores resultados en la clasificación de valoraciones positivas que clasificando las valoraciones negativas. Para resolver este

problema se desarrolla el modelo de clasificación de Naive Bayes con el que se obtiene una mejora notable.

En (Raschka, 2014) se describe el modelo de Naive Bayes aplicado a la clasificación de textos. Se explica en detalle la teoría de probabilidad condicionada, el teorema de Naive Bayes y el uso de técnicas de procesamiento de lenguaje natural (*tokenization*, *stemming*, *lemmatization*, TF, TF-IDF, etc.) para la extracción de variables que se introducen a los modelos de clasificación.

A continuación, se describen las ideas principales de los *papers* científicos en los que se ha utilizado el modelo de regresión logística. Además, en el segundo *paper* citado se utiliza también el modelo de Naive Bayes.

En (Peng Shi, 2020) se realiza la implementación del modelo de regresión logística con diferentes aproximaciones, incluyendo Descenso del Gradiente (GD), Descenso del Gradiente Estocástico (SGD) y el Descenso de Gradiente Estocástico por lotes (MBSGD). La implementación ha sido desarrollada en un entorno Big Data utilizando Hadoop y Spark, dos softwares libres para el procesamiento de datos a gran escala.

En (Vikas Khullar, 2017) se cita la aplicación de los modelos de clasificación de Naive Bayes y regresión logística para la clasificación binaria de Tweets. La solución está desarrollada en un entorno Big Data con el uso de un clúster como almacenamiento y procesamiento de los datos.

5.2 ENSEMBLE DE MODELOS

En (Nadia Felix, 2014) se cita la clasificación de Tweets en base al análisis de sentimientos utilizando el *Ensemble* de modelos de clasificación. Los experimentos descritos utilizan la salida de varios modelos para realizar la clasificación final de los Tweets. Los clasificadores utilizados son Naive Bayes, SVM, Random Forest y la regresión logística.

En (Monisha Kanakaraj, 2015) se emplea también la técnica de *Ensemble* de modelos de clasificación para compararlos con los resultados obtenidos con los modelos de clasificación aislados, realizando una clasificación de Tweets en positivos, negativos o neutros. Los modelos implementados son SVM, MaxEnt, y Naive Bayes, mientras que los modelos de *ensemble* utilizados han sido Random Forest y árboles de decisión con clasificador Ada Boost.

Por último, en (David Zimbra, 2018) se describen los resultados obtenidos tras realizar una comparación de 28 sistemas académicos desarrollados para la clasificación binaria de Tweets basada en el análisis de sentimientos. Las claves que han identificado para el análisis de sentimientos de Tweets son: los resultados de los modelos basados en el aprendizaje automático supervisado ofrecían mejores resultados que los sistemas basados en un conjunto de reglas predefinidas; los métodos de *Ensemble* de modelos ofrecen mejores resultados ya que consiguen adaptarse mejor al desequilibrio entre clases; por último se describe cómo la utilización de varios léxicos para la extracción de información consigue adaptarse mejor a los sentimientos que se pueden expresar mediante emoticonos, léxicos de negación, léxico de emociones, etc.

Capítulo 6. IMPLEMENTACIÓN MÓDULO WEB SCRAPING

Tal y como se describe en el apartado 2.2.1, para realizar un módulo de *web scraping* se suelen seguir una serie de pasos. En el presento capítulo, se detallan cada uno de los pasos seguidos para implementar el módulo de extracción de datos del proyecto y las decisiones que se fueron tomando a lo largo de su implementación.

6.1 ANÁLISIS DE LAS PÁGINAS WEB DE EMPLEO

Una vez se ha decidido que el proyecto necesita de una fuente de datos externa, el primer paso a seguir es el análisis de las distintas páginas web o de las fuentes de datos públicas que existen para conseguir la información valiosa.

Por lo tanto, se comenzó el análisis de las páginas web con más potencial de ofrecer información que sirviese para cumplir con los objetivos del proyecto. Los objetivos mencionados son la **obtención de información accionable de una web de empleo** y **el desarrollo de modelos de procesamiento de lenguaje natural para clasificar esa información**.

La primera página web que se analizó fue **Glassdoor**. Se trata de una página web estadounidense en la que antiguos y actuales empleados envían sus valoraciones sobre la empresa de una forma anónima. Además de la valoración general sobre la empresa, Glassdoor ofrece información sobre los salarios, entrevistas, beneficios y empleos de las empresas.

La información objetivo del módulo *web scraping* son las opiniones que los usuarios envían sobre las empresas para, en función de la puntuación y el texto de la valoración, poder obtener una cantidad de información suficiente para comenzar a desarrollar modelos de NLP.

El primer paso fue realizar una navegación rápida a través de la página hasta el apartado de opiniones de una empresa. Llegar hasta la información que se requiere es rápido e incluso se puede llegar a través de una URL con la terminación de la Figura 10:

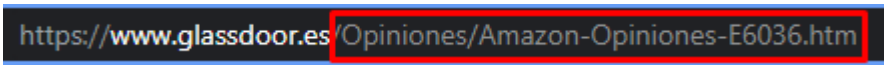


Figura 10. URL opiniones de Glassdoor.

La página web de Glassdor tiene implementada una de las técnicas de *anti-scraping* comentadas en el apartado 3.2. Esta técnica impide el acceso a la información hasta que se inicie sesión. En la Figura 11 se muestra la evidencia de la técnica:

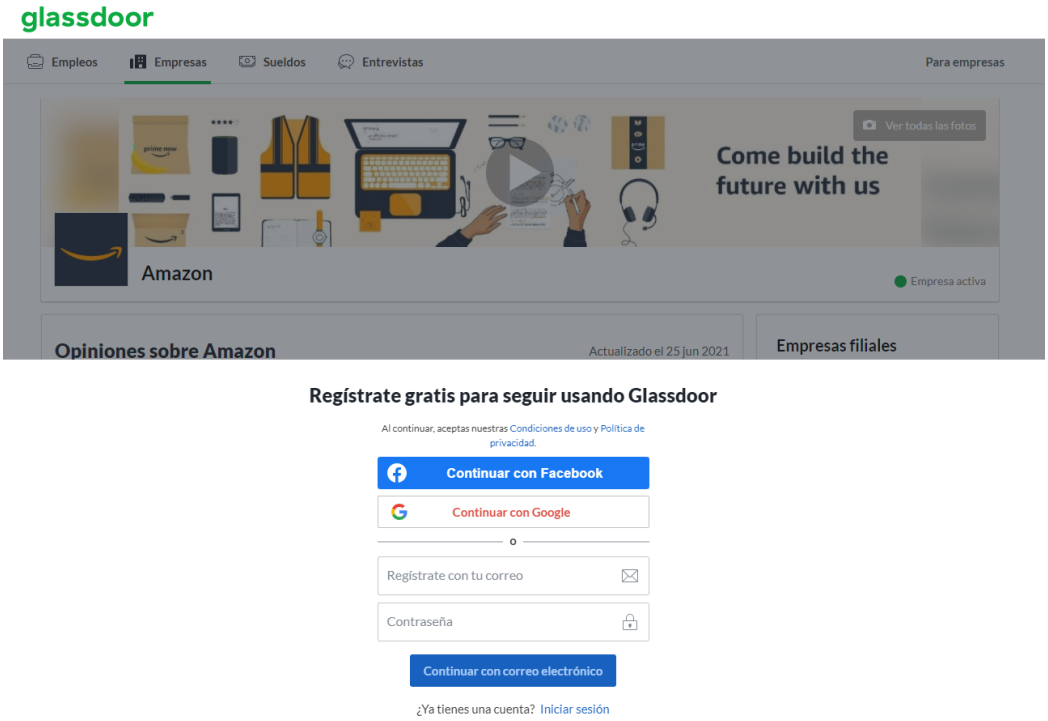


Figura 11. Registro al navegar en Glassdoor.

Este problema supone que el módulo de *web scraping* debe saber autenticarse en la página web de Glassdoor si quiere acceder a la información. Si una página web toma tales medidas de protección frente a sus datos es precisamente porque intenta evitar el *web scraping*.
Por este motivo se decidió comenzar a analizar la siguiente página web de valoraciones, **Indeed**.
Al igual que en el análisis de Glassdoor, lo primero que se realizó fue una navegación a través del navegador hasta la información valiosa para el cumplimiento de los objetivos del proyecto. Lo primero que se observa es una página en la que se muestran muchas de las empresas que tienen información dentro de la página web. Esta página puede servir de ayuda a la hora de recoger una primera información general sobre las empresas que hay y el número de valoraciones que tiene para poder empezar a tomar decisiones.

Siguiendo con la navegación, se observa que llegar hasta la página que contiene las valoraciones sobre la empresa es tan sencillo como en el caso de Glassdoor y simplemente habría que ir modificando el nombre de la empresa que se quiera extraer la información para llegar a las valoraciones de esta. En la Figura 12 se muestra la terminación de la URL para poder llegar al apartado de valoraciones.

Figura 12. URL opiniones de Indeed.

Cuando se realiza la navegación a través del listado de valoraciones que hay de cada empresa, no hay ningún impedimento para llegar a la información como ocurría en Glassdoor. Estos factores han sido muy importantes a la hora de tomar la decisión de utilizar Indeed como fuente de datos externa, pero antes de extraer la información es necesario llevar a cabo ciertos pasos importantes detallados en los siguientes apartados.

6.2 ESTUDIO DE LA LEGALIDAD

Uno de los pasos más importantes es el estudio de la legalidad a la hora de implementar un módulo de *web scraping*. Para el proyecto en concreto, la información valiosa que se quiere extraer de las páginas web se encuentra totalmente anonimizada. Esto es importante para no incumplir el Reglamento General de Protección de Datos (RGPD) [2], que protege a las personas físicas en lo que respecta al tratamiento de datos personales y a la libre circulación de estos datos.

Como se ha explicado en el apartado anterior, para poder acceder al total de las valoraciones en la página web de Glassdoor, hay que iniciar sesión y que el módulo se identifique y recoja las cookies de Glassdoor. Esto es un proceso más complejo y requiere que el usuario acepte las condiciones de uso. Por lo que se ha optado por la utilización de Indeed como fuente de datos.

Una de las posibilidades que ofrece Indeed es la utilización de su API privada para el acceso a la información publicada dentro de la página web. Esta opción ha quedado descartada puesto que se trata de una API de pago en función del uso que se vaya a hacer de la misma. En el proyecto se requiere una numerosa cantidad de valoraciones, por lo que el uso de esta API puede suponer un elevado coste.

Otros aspectos relevantes a la hora de analizar la legalidad de la implementación de un módulo *web scraping* son:

- Normativa de competencia desleal [3]: esta normativa afecta cuando el que extrae la información realiza un uso de los datos que pueda ser considerado como imitación, y pueda ser perjudicial para la otra empresa.
- Violación de los términos y condiciones de uso [4]: esta normativa afecta desde el momento en el que el usuario que navega por la página web acepta dichos términos y condiciones de uso.

6.3 ESTUDIO DE LAS POSIBLES IMPLEMENTACIONES

El siguiente paso en la realización de un módulo de *web scraping* es investigar si existe alguna API pública que facilite la obtención de la información. Esta posibilidad ha sido descartada ya que ninguno de los dos portales ofrece una API pública de acceso a sus datos.

Aunque la posibilidad de utilizar una API para la obtención de la información ha sido rechazada, existe la posibilidad de interceptar peticiones a la API privada. Este tipo de peticiones las realiza el navegador web al cargar la página web.

A continuación, se detalla cómo un módulo de *web scraping* suele interceptar este tipo de peticiones que hace una página web a un servidor para mostrar el contenido de la página web. En la Figura 13 se muestra a muy alto nivel cómo funciona este tipo de páginas.

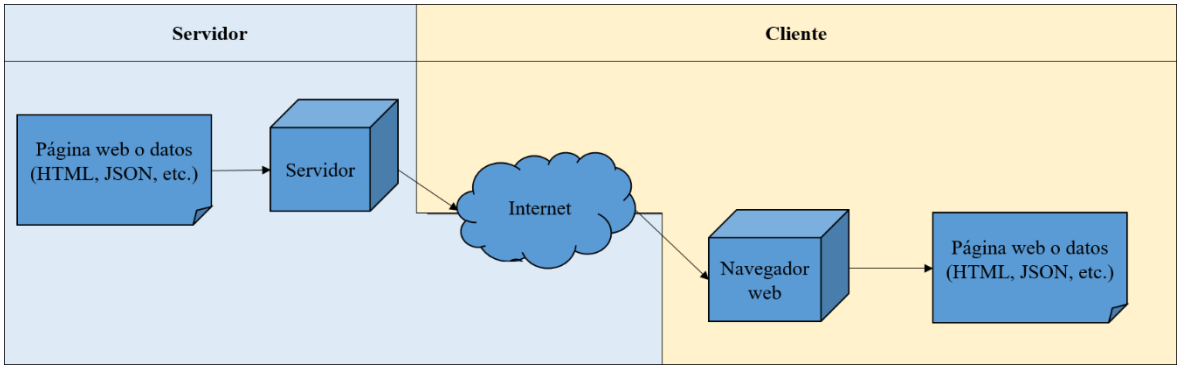


Figura 13. Petición de navegador web al servidor.

Cuando un navegador carga una página web, se realizan una serie de peticiones que en ocasiones están visibles y permiten al programador interceptar el punto final (*endpoint*) de la API a la que llama el navegador para mostrar el contenido de la página. Cuando el programador ha sido capaz de interceptar esta llamada a la API del servidor que devuelve la información de la página, puede replicar esa llamada al servidor. Como se puede observar en la imagen, la llamada a la API puede devolver la información en varios formatos. El formato más extendido y utilizado para las APIs es el formato JSON. Es el acrónimo de

notación de objetos de JavaScript y es un formato de archivo estándar y de código abierto que se utiliza para el intercambio de información que consiste en pares de clave-valor.

Realizando un análisis de la red del navegador al cargar la página web se pueden analizar las peticiones que realiza el navegador para obtener los datos con los que muestra la página web. El navegador que se ha utilizado para este proyecto ha sido un navegador Chrome, por lo que todas las capturas que se muestran pueden variar en función de la versión de Chrome instalada o del navegador que se tenga instalado. En cualquier caso, el funcionamiento de este tipo de herramientas en los navegadores suele ser muy similar.

Para acceder a la función de Chrome de inspección de la red, se accede a la sección llamada "herramientas para desarrolladores" que se encuentra en el menú desplegado de Chrome y en la sección "más herramientas". Una vez se ha abierto esta ventana, en la pestaña de "Red" se puede observar todo el tráfico que el navegador gestiona al cargar la página web.

Para analizar el tráfico que recibe un navegador web al cargar la página, se tiene que abrir la ventana esta ventana y recargar la página web. Una vez se ha realizado esto, se puede observar todo el tráfico que gestiona el navegador. En el proyecto en concreto se ha realizado con la página web de Indeed, obteniendo lo que se muestra en la Figura 14.

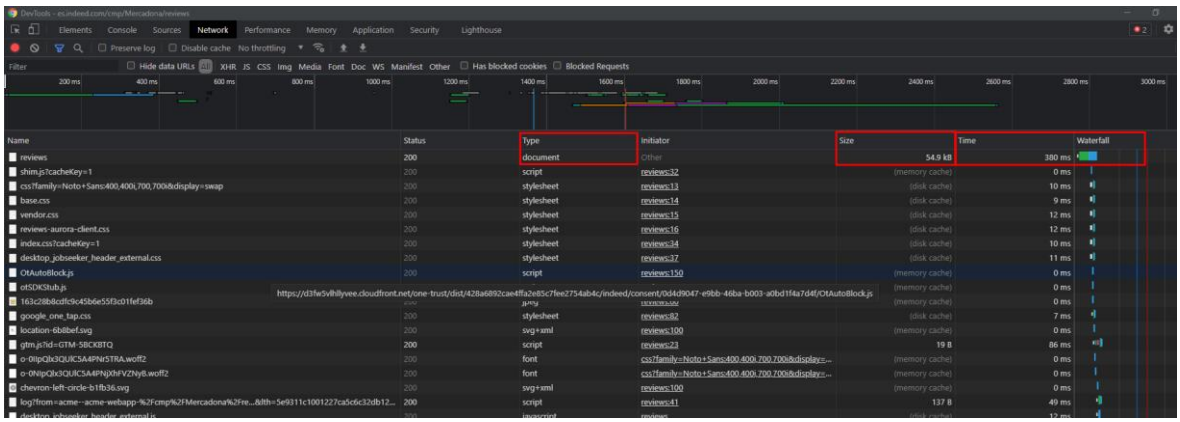


Figura 14. Herramientas para desarrollador de Chrome.

En esta sección se informa de todo el tráfico y muestra algunos valores de interés para el análisis de las peticiones relevantes para el módulo de *web scraping*. Los parámetros más importantes dentro de esta ventana son:

- Tipo: este parámetro informa del formato de la respuesta. El tipo de formato puede ser un documento, una hoja de estilo, un script, código JavaScript, etc.
- Tamaño: este parámetro informa del tamaño que tiene la información contenida en la respuesta. Lo más habitual es que la petición que contiene los datos es la que tiene un tamaño mayor.
- Waterfall: Este parámetro indica información sobre los tiempos que conlleva realizar la petición (tiempo de espera, carga, etc.). Generalmente cuanto más tiempo tarda en recibir la respuesta, más datos contiene la respuesta.

En la imagen se han destacado en rojo estos tres parámetros de una petición que ha sido de interés, ya que ocupa un tamaño de 54,9 kB, y ha sido la que más tiempo se ha tardado en procesar.

Si se analiza la petición en concreto, se puede observar la información que devuelve, la cual se muestra en la Figura 15:

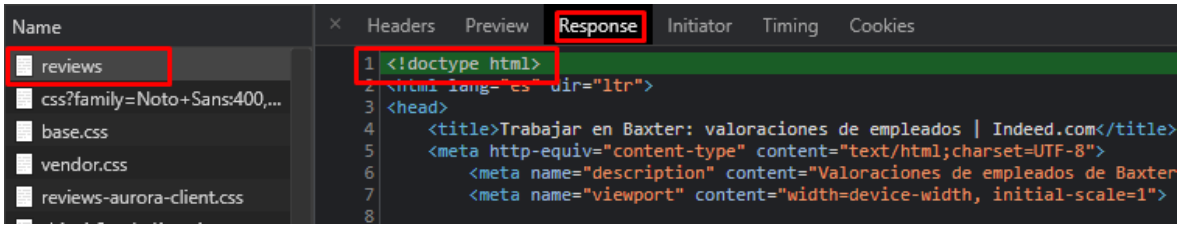


Figura 15. Captura petición con datos.

En la pestaña de respuesta, se observa el formato en el que se recibe la respuesta a esta petición. El formato es HTML, que como se ha explicado en los apartados anteriores, es un formato bastante sencillo para poder extraer la información mediante la librería BeautifulSoup. En la Figura 16 se muestra la petición que realiza el navegador para obtener la información que muestra por pantalla.

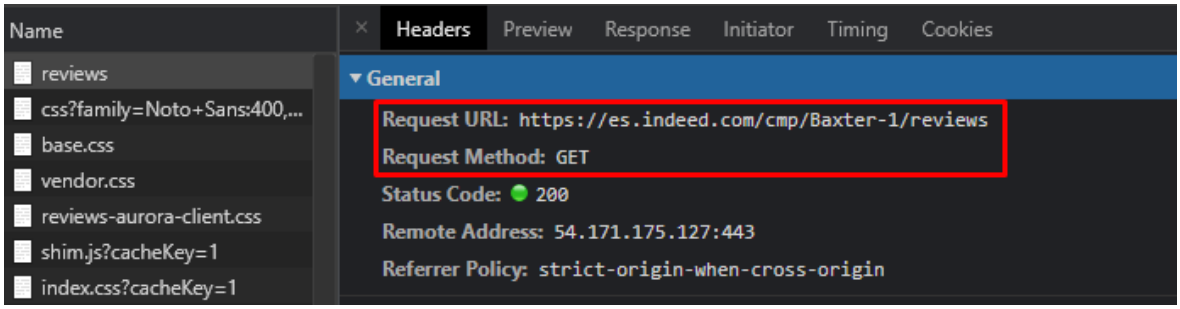


Figura 16. Cabecera de la petición request a la información de Indeed.

Por lo tanto, el siguiente paso es comprobar que se puede replicar esa petición en Python. En la Figura 5 se muestra un ejemplo realizado con la librería Requests. De esta forma, se puede confirmar que, si realizamos peticiones a las URLs de las valoraciones y realizamos el procesamiento de la información del archivo HTML de la respuesta, se puede obtener la información.

Esta es una de las implementaciones llevadas a cabo en el módulo de *Web scraping* cuyo detalle del proceso de desarrollo se muestra en el apartado 6.5.

Sin embargo, uno de los objetivos secundarios es la exploración de las posibles vías a la hora de desarrollar un módulo de *web scraping*. Una de las librerías más potentes y útiles para hacer *web scraping* es Selenium, y la segunda forma de extraer la información se ha centrado en realizarla con esta librería de Python.

Según lo mencionado en el apartado 3.3.3, Selenium permite al programador tomar el control de un navegador por medio de un script de Python. En la Figura 7 (Capítulo 3.) se muestra un ejemplo de script usando Selenium.

En resumen, las formas desarrolladas para extraer la información han sido mediante la programación HTTP (librerías Requests y BeautifulSoup) y mediante las técnicas de control de navegador web basado en Selenium (librería Selenium).

6.4 *ESTRUCTURA DE LOS MÓDULOS*

La estructura desarrollada para los módulos sigue la explicada en el apartado 2.2.1 y que se muestra en la Figura 1. El primer módulo que se ha desarrollado es el conocido como ***spider***, que intenta recoger la mayor información posible de las páginas web, siendo normalmente la recopilación de nuevas URLs el objetivo de este módulo.

En el proyecto, este módulo se ha dividido en dos partes principales y con un objetivo bien definido. Las partes que se han desarrollado han sido las siguientes:

- **Parte 1**: el objetivo principal es conseguir recopilar el mayor número de empresas que tienen información en la página web de Indeed. Para ello, el *spider* ha extraído la información de la página de inicio de las valoraciones de empresa. En esta página se muestran 24 empresas cada vez que se carga esta página, de la que se puede extraer el nombre de la empresa, el número de valoraciones que tiene, la puntuación que tiene total (sobre cinco) y lo más importante, la URL que lleva a la página sobre la empresa. En la Figura 17 se muestra un ejemplo de la página extraída por el *spider*. Este paso se ha realizado iterando sobre la página de inicio y se ha construido un fichero "empresas.json" con información general sobre 90 empresas.

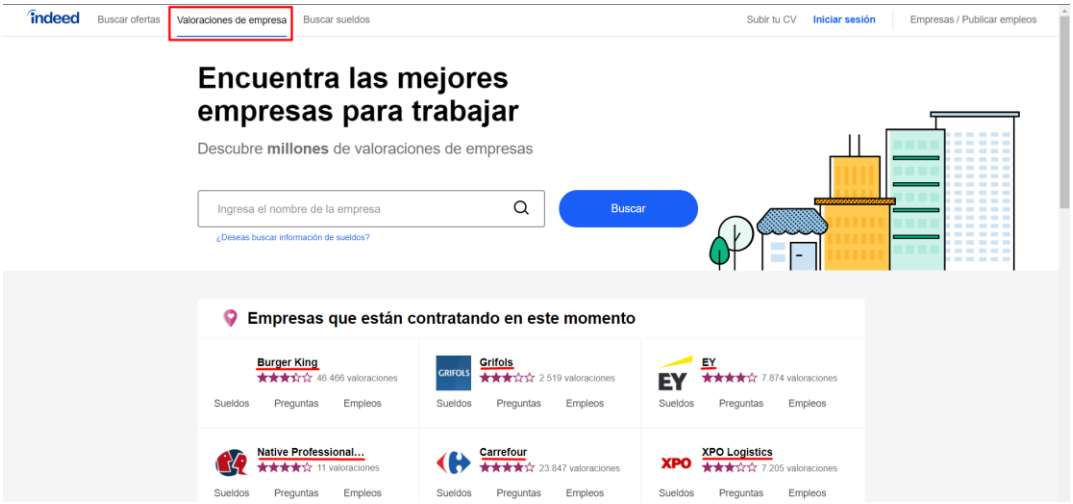


Figura 17. Página extraída por el spider.

Parte 2: el objetivo de esta segunda parte del *spider* es completar la información que ha generado la primera parte. Simplemente se ha desarrollado una función que extrae la información que hay en la página de inicio de cada empresa. La información que se ha añadido en el fichero de "empresas.json" ha sido el año de fundación y el resumen de la empresa, en la

- Figura 18 se muestra un ejemplo de fichero JSON generado que contiene una valoración de una empresa. Además, en la Figura 19 se muestra la página de inicio de una empresa donde se puede encontrar esa información.


```
{
    "url": "https://es.indeed.com/cmp/empresa",
    "nombre": "empresa",
    "num_valoraciones": 20,
    "punt_total": 4,
    "fundacion": "Lorem ipsum...",
    "resumen": "Lorem ipsum...",
    "valoraciones": {
        "1": {
            "nota_val": 2,
            "array_cat": [
                "width:15px",
                "width:27px",
                "width:15px",
                "width:15px",
                "width:63px"
            ],
            "tex_titulo": "Lorem ipsum...",
            "tex_valoracion": "Lorem ipsum...",
            "tex_ventajas": "Lorem ipsum...",
            "tex_contras": "Lorem ipsum...",
            "url_valoracion": "https://es.indeed.com//cmp/
            "list_autor": [
                "Consultor (Antiguo empleado) ",
                " Madrid ",
                " 14 enero 2017"
```


Figura 18. Ejemplo del fichero JSON "empresas.json".

Figura 19. Página de inicio de una empresa.

Estas partes se han desarrollado por separado, tanto con la programación HTTP, como con la librería Selenium.

El bloque más importante, donde se recoge la mayor parte de la información, es el ***scraper*** que recopila las valoraciones y la información disponible dentro de la pestaña de valoraciones de cada empresa que se quiera extraer. En la Figura 20 se muestra un ejemplo de cómo es la página de las valoraciones de una empresa.

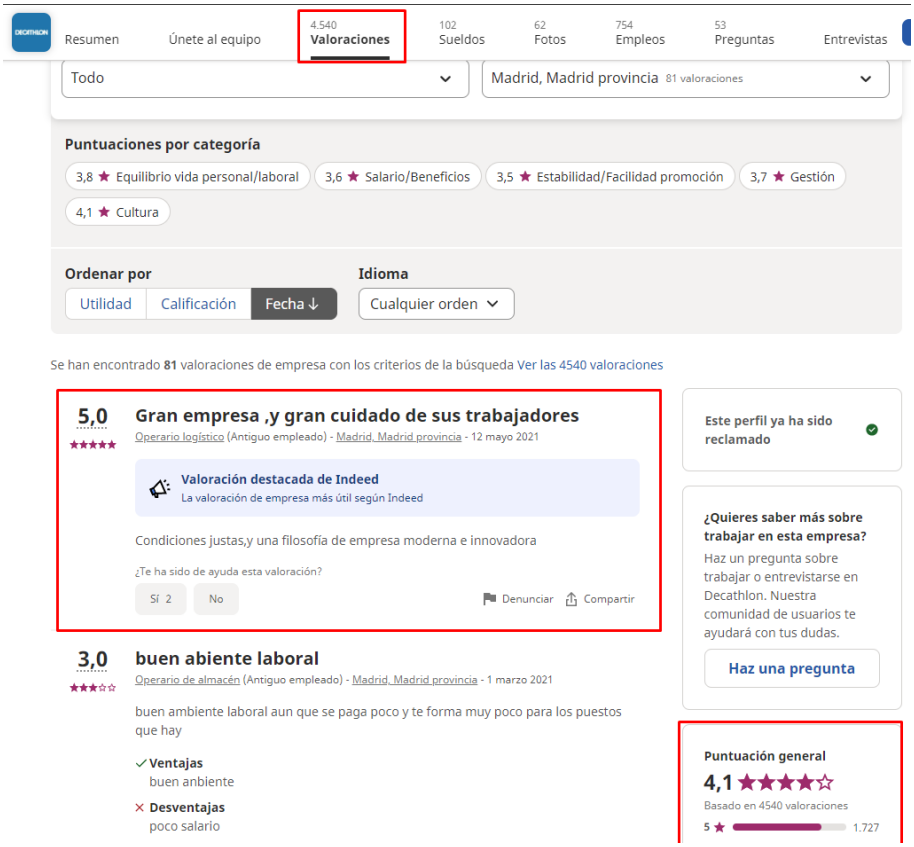


Figura 20. Página de valoraciones de una empresa.

A diferencia del bloque con el *spider*, este bloque no se divide en dos partes. Sin embargo, tiene dos tareas que se pueden diferenciar claramente:

- **Tarea 1**: esta tarea se centra en la extracción de la información de las valoraciones en conjunto. Recoge la información de las categorías disponibles a evaluar en cada valoración, algunos enlaces a valoraciones por puesto de trabajo y algunos enlaces a valoraciones por localidad.

  Esta información no ha sido utilizada posteriormente, pero puede ser de utilidad para analizar las valoraciones dependiendo de cada localidad, puesto, categoría, etc.
- **Tarea 2**: esta es la tarea principal de este bloque y consiste en la recopilación de la información de cada valoración. Los campos que extrae de cada valoración son los siguientes:

- Título.
- Texto de la valoración.
- Texto de ventajas.
- Texto de desventajas.
- Nota de la valoración.
- Nota de las categorías: Indeed tiene cinco categorías definidas para valorar (Equilibrio vida personal/laboral, Remuneración/Beneficios, Estabilidad laboral/Facilidad de promoción, Gestión y Cultura).
- Puesto.
- Localidad.
- Fecha de la valoración.
- Votación útil: si la valoración le ha resultado de utilidad a otra persona.
- Votación no útil: si la valoración no le ha resultado de utilidad a otra persona.

Este bloque se ejecuta individualmente por empresa o por una lista de empresas que recibe como parámetro.

Por último, se ha realizado el bloque conocido como ***parser***. Este se ha centrado en el procesamiento de los ficheros extraídos por el *scraper* para generar un fichero que tenga la información del *scraper* bien ordenada y colocada según los campos explicados anteriormente. Las tareas que realiza son las siguientes:

- Eliminación de registros **duplicados**: para asegurar que no hay registros duplicados.
- Extracción de la **fecha de publicación**: el fichero JSON generado almacena la fecha junto con el puesto y con la localidad.
- Extracción de las **notas por categoría**: el fichero JSON generado almacena las notas de las categorías en forma de lista.
- Extracción de las **votaciones de utilidad y no utilidad**.
- Extracción del **idioma** de la valoración: Indeed no proporciona la información sobre el idioma en el que está escrita, con lo que esta tarea cobra especial importancia.

- Separación de las valoraciones en dos ficheros JSON (valoraciones en inglés y valoraciones en español) para poder trabajar con las valoraciones en inglés en la segunda parte de proyecto.

6.5 DESARROLLO DE LA IMPLEMENTACIÓN CON BEAUTIFULSOUP

En este apartado se describe la lógica de las funciones implementadas en los bloques del *spider* y el *scraper* que se han explicado en el apartado anterior.

6.5.1 DESARROLLO DEL SPIDER

A continuación, se detalla la lógica implementada en la parte 1 del bloque. Los pasos que sigue la función implementada para la parte 1 son los siguientes:

1. Lectura de los datos del fichero JSON ("empresas.json") con la información de las empresas ya guardadas, de esta forma el *spider* actualiza la información existente o inserta una nueva empresa en caso de que no se haya encontrado todavía.
2. Creación de la variable "url" que tiene la dirección URL de la página de inicio de valoraciones de empresas.
3. Creación de un bucle que se ejecuta el número de veces que se le pasa cómo parámetro a la función (número de veces que se quiere recargar la página). En este bucle se realiza lo siguiente:
    a. Petición HTTP con la librería Requests.
    b. Creación de un elemento de BeautifulSoup con la respuesta de la petición que se conoce que es un archivo HTML.
    c. Extracción de los elementos de BeautifulSoup que tienen la información de cada empresa.
    d. Bucle de extracción de la información de cada elemento.
4. Actualización y guardado del fichero "empresas.json" con la nueva información.

Por lo tanto, el script ejecuta tantas peticiones HTTP – GET como número de iteraciones se le pase a la función.

Uno de los elementos clave de este bloque es formar la lista de 24 empresas que carga la página de inicio cuando se realiza la petición HTTP y se crea el elemento de BeautifulSoup. En la Figura 21 se muestra la línea del código de la función para poder extraer dicha lista.

```
list_divs_empresas = soup.find_all(class_="cmp-PopularCompaniesWidget")
```

Figura 21. Extracción lista de 24 empresas página de inicio.

En esta línea de código se ejecuta la función llamada "find_all" de BeautifulSoup que encuentra todos los elementos HTML cuyo nombre de clase sea en indicado como parámetro. En la Figura 22 se puede observar el nombre de la clase del elemento HTML que contiene la información a extraer "cmp-PopularCompaniesWidget".

Figura 22. Elemento HTML que recoge la información de la empresa (página de inicio).

La clase se refiere al estilo que se le ha aplicado al elemento HTML y que se le pasa como un atributo.

En cuanto a la segunda parte, la función realiza los siguientes pasos:

1. Lectura de los datos del fichero JSON ("empresas.json").
2. Creación de un bucle que realiza lo siguiente por cada empresa del fichero JSON:

a. Petición HTTP a la página de inicio de la empresa.
b. Extracción de la información nueva (año de fundación y resumen) y actualización de la antigua.

3. Actualización y guardado del fichero "empresas.json".

6.5.2 DESARROLLO DEL SCRAPER

Tal y como se ha descrito en el apartado 6.4, el *scraper* realiza principalmente dos tareas. La primera tarea (extracción de información general de las valoraciones de la empresa) se realiza al comienzo del script y sólo se ejecuta una vez, mientras que la segunda tarea (extracción de la información del bloque de valoración) se ejecuta para cada una de las valoraciones que tiene cada empresa de la que se quiera extraer información.

Un aspecto importante en la implementación de esta función es la lógica que sigue para la extracción de todas las valoraciones de la empresa. Si una empresa tiene 1.000 valoraciones y en cada página de valoraciones se muestran 20 valoraciones, entonces el script debe realizar 50 peticiones HTTP a cada una de las páginas correspondientes para la extracción de todas las valoraciones. Para ello, es necesario analizar cómo cambia la URL de Indeed con el paso de las páginas para realizar de forma correcta la petición. En la Figura 23 se muestra cómo se forma la URL a medida que se van pasando las páginas.

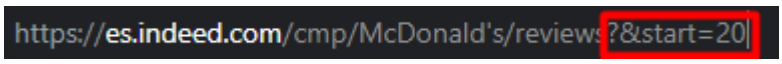


Figura 23. URL página 2 de valoraciones.

Cómo se puede observar, lo único que hay que añadir es el parámetro "start" en la petición HTTP y de esta forma se carga la página 2 de valoraciones.

Una vez descrita la forma de navegar entre las distintas páginas, a continuación, se detallan los pasos que realiza el script:

1. Lectura del fichero o lista de empresas que se quieran extraer sus valoraciones.
2. Creación de un bucle por cada empresa, que realiza lo siguiente:

a. Creación de una lista de enteros con los números que habría que pasarle al parámetro "start" en la petición (Ejemplo: 1.000 valoraciones, entonces la lista de enteros sería [0, 20, 40, 60, …, 980]).
b. Creación de la lista de URLs a partir de la lista de enteros obtenida.
c. Creación de un bucle por cada URL que realiza lo siguiente:
    i. Petición Requests a la URL.
    ii. Si es la primera URL, entonces se realiza la tarea 1 de extracción de información general.
    iii. Extracción de las valoraciones de la página correspondiente (20 valoraciones por página).

3. Guardado de un fichero JSON por cada empresa.

Para la extracción de las valoraciones por cada página se ejecuta la misma función que se ha mostrado en la Figura 21, pero con el nombre de la clase "css-lqffld-Box eu4oa1w0". En la Figura 24 se muestra como se ha encontrado el nombre de la clase.

Figura 24. Elemento HTML con la información de cada valoración.

6.6 DESARROLLO DE LA IMPLEMENTACIÓN CON SELENIUM

En este apartado se va a explicar la lógica de las funciones implementadas en los bloques del *spider* y el *scraper* que se han explicado en el apartado 6.4.

6.6.1 DESARROLLO DEL SPIDER

La lógica implementada en la primera parte es muy similar a la implementada con la programación HTTP explicada en el apartado anterior, pero cambian algunos detalles que se explican en la siguiente lista de pasos realizados por la función:

1. Lectura de los datos del fichero JSON ("empresas.json").
2. Creación de la variable "url" de la página de inicio.
3. **Comienzo del control del navegador de Chrome por parte de Selenium**.
4. Creación de un bucle que se ejecuta el número de veces que se le pasa cómo parámetro a la función. En este bucle se realiza lo siguiente:
    a. Carga de la URL de inicio (tiempo de espera).
    b. Extracción de los **elementos de Selenium** que tienen la información de cada empresa. Estos elementos se obtienen a partir del **driver de Chrome** que controla el navegador.
    c. Creación de un bucle para extraer la información de cada elemento.
5. Actualización y guardado del fichero "empresas.json" con la nueva información.

En la Figura 21 se muestra la línea de código que se utiliza con la librería BeautifulSoup para la extracción de la lista de 24 empresas que se cargan en la página de inicio. La línea de código que realiza la misma función en la librería Selenium, se muestra en la Figura 25.

```
list_divs_empresas = driver.find_elements_by_class_name("cmp-PopularCompaniesWidget")
```

Figura 25. Extracción lista de 24 empresas página de inicio (Selenium).

En cuanto a la segunda parte, la función realiza lo siguiente:

1. Lectura de los datos del fichero JSON ("empresas.json").
2. **Comienzo del control del navegador de Chrome por parte de Selenium**.
3. Creación de un bucle que realiza lo siguiente por cada empresa del fichero JSON:
    a. Carga de la URL en el navegador (tiempo de espera).
    b. Extracción de la información nueva (año de fundación y resumen) y actualización de la antigua.
4. Actualización y guardado del fichero "empresas.json".

6.6.2 DESARROLLO DEL SCRAPER

El desarrollo del *scraper* con Selenium tiene más diferencias con respecto al *spider*, principalmente en la forma de navegar por las distintas páginas de las valoraciones. En el *scraper* con la programación HTTP es necesario calcular con anterioridad el número de peticiones que había que hacer para obtener todas las valoraciones y formar las URLs de cada página. Sin embargo, con Selenium esto no es necesario puesto que se puede saber cuándo se ha llegado al final por medio de los botones de paginación que hay. En la Figura 26 se muestra un ejemplo de los botones mencionados.

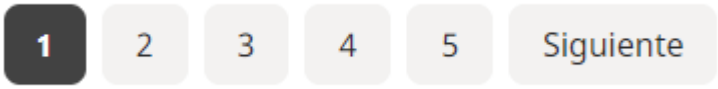


Figura 26. Botones de paginación de las valoraciones.

Si no existe el botón siguiente, es porque se ha llegado al final de las valoraciones y no hace falta seguir extrayendo más información.

Una vez descrita la forma de navegar entre las distintas páginas, a continuación, se detallan los pasos que realiza el script:

1. Lectura del fichero o lista de empresas que se quieran extraer sus valoraciones.
2. Comienzo del control del navegador de Chrome por parte de Selenium.
3. Creación de un bucle por cada empresa, que realiza lo siguiente:

a. Carga de la URL de la primera página de valoraciones (tiempo de espera).
b. Realización de la tarea 1 de extracción de información general de las valoraciones.
c. Ejecución de una función que detecta si hay botón "siguiente".
d. Extracción de las valoraciones de la primera página.
e. Creación de un bucle (while siguiente == True) que se ejecuta si la función de anterior ha detectado que hay botón "siguiente", y por lo tanto hay más valoraciones. El bucle realiza lo siguiente:
    i. Clic en el botón siguiente que se ha detectado (tiempo de espera).
    ii. Ejecución de la función de detección del botón "siguiente".
    iii. Extracción de las valoraciones de la página nueva cargada.
    iv. Cuando la variable siguiente deja de ser "True", es porque no hay botón siguiente y se termina el bucle anterior.

4. Guardado de un fichero JSON por cada empresa.

El código que ejecuta el paso "i" explicado en la lista de pasos se muestra en la Figura 27.

```
driver.execute_script("arguments[0].click();", boton_siguiente)
```

Figura 27. Clic en el botón siguiente con la función execute_script de Selenium.

Las acciones que realiza un usuario cuando controla el navegador (clic en botones, escribir en los cuadros de texto o hacer *scroll*[5] por la página web) son acciones que se realizan con código JavaScript mediante la activación de ciertos eventos que modifican la página web cuando se detectan. Esta posibilidad de interactuar es una de las grandes ventajas que ofrece Selenium y que son posibles con la ejecución de comandos como el que se muestra en la Figura 27.

[5] Mover el texto o los gráficos mostrados en una dirección determinada en la pantalla de un ordenador para ver diferentes partes de los mismos.

Esta solución debe tener en cuenta los tiempos de carga de un navegador, por lo que se tarda más que con la programación HTTP. En los pasos en los que se indica "tiempo de espera" es donde se han introducido unos segundos para que le dé tiempo al navegador a cargar la nueva página.

Capítulo 7. DESARROLLO DE LOS MODELOS DE NLP

En el siguiente capítulo se describen los pasos llevados a cabo en el proyecto para el desarrollo de los modelos de clasificación de las valoraciones.

Puesto que el estudio de las valoraciones no se ha dirigido a un sector empresarial concreto, se ha decidido obtener las valoraciones de empresas de diferentes sectores. Los sectores elegidos han sido los siguientes:

- Consultoría.
- Transporte.
- Turismo.
- Alimentación.

Con esta decisión se consigue un vocabulario más extendido y más variedad en la puntuación de las valoraciones.

En el primer entrenamiento de los modelos de clasificación se hizo uso sólo del texto contenido en la valoración, pero en los siguientes entrenamientos se fueron introduciendo los textos correspondientes al título de la valoración, a las ventajas y a las desventajas. Esto ha aportado más información y ha permitido mejorar las métricas de los modelos.

7.1 PREPROCESADO DEL TEXTO EXTRAÍDO DE INDEED

En este apartado se describen las transformaciones realizadas a las valoraciones extraídas de Indeed, para convertirlas en variables aprovechables por los modelos de clasificación desarrollados.

Las valoraciones que se han utilizado para entrenar los modelos de clasificación han sido las valoraciones que estaban escritas en inglés. En esta decisión han influido dos motivos:

- El número de valoraciones en inglés era mayor: Tras haber analizado los datos de salida del procesado realizado en el *parser*, se habían obtenido 99.484 valoraciones en inglés, frente a las 9.820 de español.
- El desarrollo de las librerías de NLP utilizadas está más avanzado para el procesamiento de textos en inglés.

Como se ha explicado en el apartado 6.4, el procesamiento final separa las valoraciones en dos idiomas principales (valoraciones en inglés y valoraciones en español). Esta separación se ha realizado haciendo uso de la librería de Python llamada *langdetect*. En la Figura 28 se muestra un ejemplo de la librería.

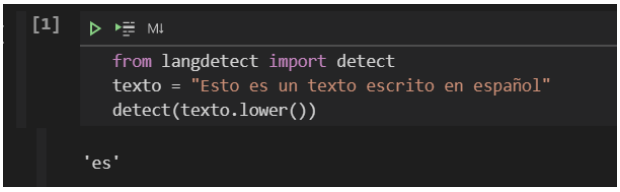


Figura 28. Ejemplo de librería langdetect.

Por lo tanto, los datos de entrada de los modelos son las valoraciones escritas en inglés.

Para poder entrenar los modelos de clasificación binaria se ha realizado la separación de las valoraciones en valoraciones positivas y valoraciones negativas. Para ello, se ha elegido la nota general de la valoración como umbral para separar en estas dos clases. Las valoraciones que tienen una puntuación de 1 o 2 se han clasificado como malas (valor 0) y las valoraciones con la puntuación de 4 o 5, se han clasificado como buenas (valor 1). En la Figura 29 se muestra la distribución de las valoraciones obtenidas según la nota general de la valoración.

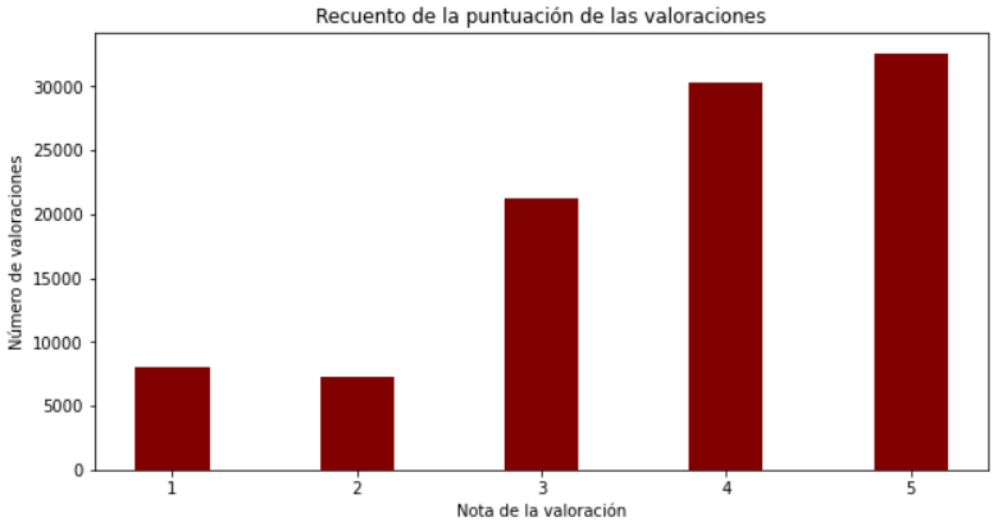


Figura 29. Distribución de las notas de las valoraciones.

Se puede observar que el número de valoraciones positivas es mayor que de las valoraciones negativas, por lo que se tendrá un conjunto de datos desbalanceado. Una vez se ha realizado la clasificación binaria (valoraciones buenas y malas), quedan eliminadas las valoraciones con una puntuación de 3 y un porcentaje de valoraciones malas del 25%. En la Figura 30 se muestra la distribución final del conjunto de valoraciones.

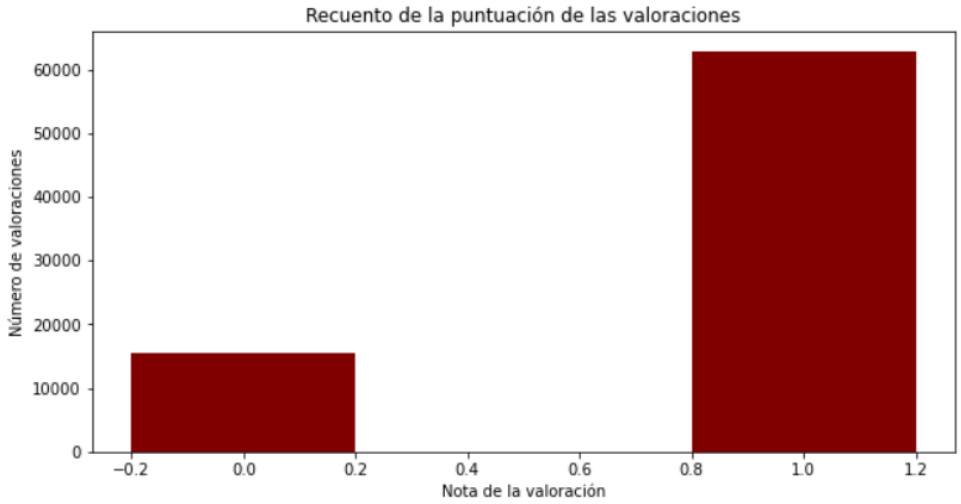


Figura 30. Distribución de las valoraciones final (clasificación binaria).

Tras la preparación del conjunto de valoraciones a utilizar para el desarrollo del modelo, se desarrolló el bloque de procesado del texto de cada valoración. Como se ha descrito en el apartado 4.3.1 las técnicas que se han utilizado han sido la *tokenization* y el *stemming*.

Tras realizar un primer análisis de las valoraciones extraídas, se han observado que algunas contenían hipervínculos a otras páginas web. Esta información no ha sido introducida a los modelos por la posible falta de interpretabilidad, para ello, se ha utilizado una expresión regular que elimina cualquier tipo de enlace. En la Figura 31 se muestra la línea de código que realiza esta operación.

```
# Eliminación de hyperlinks
valoracion2 = re.sub(r'https?:\/\/.*[\r\n]*', '', valoracion)
```

Figura 31. Eliminación de hipervínculos.

Los modelos que se han desarrollado utilizan el diccionario de frecuencias para la fase de entrenamiento. Para poder desarrollar un diccionario de frecuencias, el primer paso a realizar es la *tokenization* de las valoraciones. Los fragmentos en los que se ha dividido cada valoración han sido las palabras. Para ello, se ha utilizado la librería NLTK, en concreto la función TweetTokenizer que permite la separación de emoticonos, los cuales son de mucha utilidad ya que expresan de forma muy clara la positividad o negatividad de una valoración. En la Figura 32 se muestra las líneas de código que realizan el "tokenizado".

```
# Tokenizado
tokenizer = TweetTokenizer(preserve_case=False, strip_handles=True, reduce_len=True)
val_tokens = tokenizer.tokenize(valoracion2)
```

Figura 32. Tokenizado de las valoraciones.

En el lenguaje humano hay muchas palabras que no aportan información a la hora de analizar el sentimiento de una frase, estas palabras son conocidas como *stopwords*. En el diccionario de frecuencias construido para que los modelos realicen el entrenamiento no es recomendable introducir este tipo de palabras porque puede llevar a resultados erróneos. Por lo tanto, el siguiente paso que se ha realizado ha sido la eliminación de las *stopwords*. Se ha

hecho uso de la lista de palabras *stopwords* que tiene la librería NLTK para el inglés. En la Figura 33 se muestra el fragmento de código que elimina este tipo de palabras de la lista de tokens.

```
vals_clean = []
for palabra in val_tokens:
    if (palabra not in stopwords_english and palabra not in string.punctuation):
        vals_clean.append(palabra)
```

Figura 33. Eliminación de las stopwords.

El último paso para la realización del procesamiento al completo de las valoraciones ha sido la implementación de la técnica explicada en el apartado 4.3.1.2, el *stemming*. Para ello, se ha utilizado la función *PorterStemmer* de la librería NLTK. Este algoritmo de reducción de la palabra a su tallo sólo es utilizable para el inglés y se caracteriza por su rapidez y simplicidad a la hora de eliminar los prefijos y sufijos utilizados en ese idioma. En la Figura 34 se muestra el fragmento de código que se ha desarrollado para realizar el *stemming* de las palabras.

```
# Stemming
stemmer = PorterStemmer()
vals_stem = []
for palabra in vals_clean:
    stem = stemmer.stem(palabra)
    vals_stem.append(stem)
```

Figura 34. Stemming de los tokens.

El proceso completo consigue transformar una valoración en una lista de palabras que tienen significado dentro de la valoración, y en su forma más abreviada (tallo). Este procesamiento es muy importante para poder realizar un diccionario de frecuencias de las palabras que aparecen en las valoraciones que contenga la información que les resulta de mayor utilidad a los modelos de clasificación.

En la Figura 35 se muestra un ejemplo de cómo quedaría una breve valoración tras haber sido procesada por la función implementada.

```
texto = 'Hopefully , one day, I will have the opportunity again to work in Mercadona'
print(process_val(texto))
```

```
['hope', 'one', 'day', 'opportun', 'work', 'mercadona']
```

Figura 35. Procesamiento completo de una valoración.

Tras haber realizado la función de procesamiento de las valoraciones, se desarrolló la función que construye un diccionario de frecuencias. Este diccionario almacena el número de veces que una palabra ha aparecido en una valoración positiva, y el número de veces que esa misma palabra ha aparecido en una valoración negativa. Para desarrollar esta función se han seguido los siguientes pasos:

1. Creación del diccionario de frecuencias vacío.
2. Lectura de la lista de valoraciones y la lista de etiquetas de esas valoraciones (si pertenece a una valoración positiva o negativa).
3. Creación de un bucle que ejecuta la función process_val.
4. Creación de un bucle que recorre la lista de palabras que devuelve la función process_val.
    a. Comprobación de que el par (palabra, etiqueta) está en el diccionario para ir sumando al diccionario de frecuencias. Si no se encuentra en el diccionario, se añade.
5. Devolución del diccionario formado por el conjunto de valoraciones que ha recibido como parámetros.

7.2 REGRESIÓN LOGÍSTICA

Para la realización del modelo de regresión logística, ha sido necesario la implementación de las funciones explicadas en el apartado 4.3.2. En concreto se han implementado las siguientes funciones:

- Sigmoide.
- Descenso de gradiente.
- Extracción de *features*: función que realiza la extracción de características de una valoración para poder introducir estas características al modelo. Realiza el procesado de la valoración y devuelve un vector con la positividad y la negatividad de la frase.
- Función de entrenamiento del modelo.
- Función de test del modelo.

7.2.1 ENTRENAMIENTO DEL MODELO

Para el entrenamiento se ha utilizado el conjunto de valoraciones en inglés con clasificación binaria que se ha mostrado en la Figura 30. Este conjunto se ha dividido en dos, un conjunto de entrenamiento que contiene el 80% de las valoraciones y un conjunto de test que contiene el 20% restante. La división se ha realizado de tal forma que se mantiene la misma proporción de valoraciones positivas y negativas en los conjuntos de entrenamiento y de test. Tras haber dividido el conjunto de valoraciones, se realiza la construcción del diccionario de frecuencias del conjunto de entrenamiento.

En cuanto al entrenamiento del modelo, se han utilizado el descenso de gradiente y la función de coste. Los pasos realizados para el entrenamiento del modelo son los siguientes:

1. Creación de una matriz inicializada a 0 de dimensiones nxm. Siendo n el número de valoraciones del conjunto de entrenamiento y m el número de variables introducidas al modelo (más una que es el *bias*[6]).

[6] Se define como el fenómeno de observar resultados sistemáticamente perjudicados por supuestos erróneos.

2. Creación de un bucle sobre el conjunto de valoraciones: extrae las características del conjunto de entrenamiento y lo guarda en la matriz.
3. Entrenamiento utilizando el descenso de gradiente. Haciendo uso de una tasa de aprendizaje de $1*10^{-9}$ y 1.500 iteraciones para completar el entrenamiento.

Cuando se ha terminado la ejecución de la función descenso de gradiente, se obtiene un vector de pesos θ.

En el presente proyecto se han entrenado tres modelos diferentes, el primero de ellos sólo ha utilizado el texto de las valoraciones. Tras realizar el entrenamiento del modelo se obtuvo el siguiente vector de pesos θ, que indica la importancia y el efecto de cada una de las variables a utilizar en la regresión logística. La positividad del texto se calcula accediendo a la puntuación que tiene cada palabra en el diccionario de frecuencias positivo, y la negatividad del texto se calcula accediendo a la puntuación que tiene cada palabra en el diccionario de frecuencias negativo.

Bias	Positividad texto	Negatividad texto
7e-08	6.946e-05	-0.00027495

Tabla 2. Pesos del modelo 1 regresión logística

El segundo de los modelos entrenados ha utilizado el texto del título junto con el texto de la valoración (concatenados). En la Tabla 3 se muestran los valores del vector de pesos θ.

Bias	Positividad título + texto	Negatividad título + texto
7e-08	9.264e-05	-0.00039371

Tabla 3. Pesos del modelo 2 regresión logística.

Por último, se entrenó un modelo que introducía el texto correspondiente a las ventajas y el texto correspondiente a las desventajas. Para ello, se han introducido ambos textos como nuevos predictores del modelo de regresión y se han calculado sus diccionarios de frecuencias del conjunto de entrenamiento para ambos (ventajas y desventajas). El vector de pesos obtenido ha sido el siguiente:

Bias	Pos título + texto	Neg título + texto	Pos ventajas	Neg ventajas	Pos desventajas	Neg desventajas
3.6e-07	5.223e-05	- 0.000128	1.619e-05	8.96e-06	0	0

Tabla 4. Pesos del modelo 3 de regresión logística.

Realizando una visualización de la matriz de características del primer modelo entrenado, sobre los ejes de positividad y negatividad, se obtiene lo que se muestra en la Figura 36.

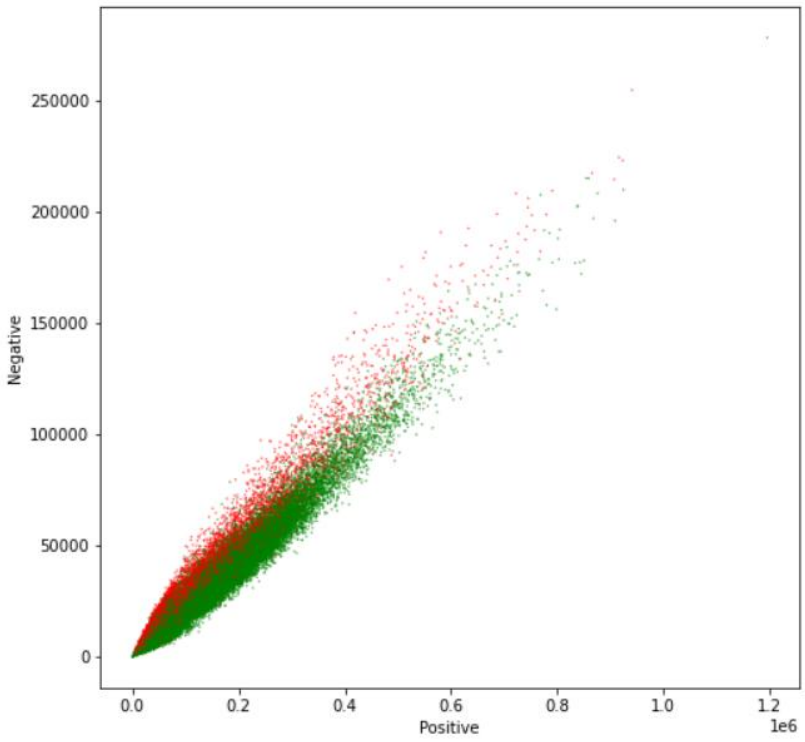


Figura 36. Visualización valoraciones modelo 1 (Texto de las valoraciones).

Sin embargo, este tipo de visualización no se puede realizar para el tercer modelo, ya que hay más predictores y al realizar la visualización en dos dimensiones no se puede realizar una buena interpretación de la visualización.

7.2.2 VALIDACIÓN SOBRE EL CONJUNTO DE TEST

Para la obtención de las métricas modelo, se ha implementado una función de test que realiza las predicciones del conjunto de test. Los resultados obtenidos se detallan en el Capítulo 8.

7.3 NAIVE BAYES

Para la construcción del modelo de clasificación de Naive Bayes, ha sido necesario la implementación de las funciones descritas en el apartado 4.3.3. En concreto se han implementado las siguientes funciones:

- Función de entrenamiento del modelo: función que contiene todos los cálculos de probabilidades necesarios para el cálculo del log Likelihood.
- Función de test del modelo.

7.3.1 ENTRENAMIENTO DEL MODELO

Para el entrenamiento se ha utilizado el mismo método que para la regresión logística. Se han utilizado las valoraciones en inglés con la clasificación binaria y se ha dividido el conjunto de valoraciones en 80% entrenamiento y 20% test. El método de creación del conjunto de entrenamiento y test ha sido el mismo en ambos modelos, por lo que los conjuntos son idénticos. Del mismo modo que para la regresión logística, el siguiente paso es la construcción del diccionario de frecuencias del conjunto de entrenamiento.

En cuanto al entrenamiento del modelo, se ha empleado el procedimiento detallado en el apartado 7.3.1, para obtener un diccionario con todo el vocabulario de entrenamiento y el valor del log Likelihood de cada palabra. Los pasos llevados a cabo para el entrenamiento del modelo son los siguientes:

1. Inicialización del diccionario log Likelihood.
2. Bucle que recorre todas las palabras del diccionario de frecuencias para calcular las probabilidades que permiten el cálculo del log Likelihood de cada palabra.

Cuando se completa el entrenamiento del modelo se obtiene el diccionario con el valor del log Likelihood de todas las palabras del conjunto de entrenamiento y el valor de la probabilidad a priori (indica el desbalanceo del conjunto de entrenamiento).

Para realizar una comparación entre los resultados del modelo de clasificación utilizando regresión logística o Naive Bayes, se ha realizado el entrenamiento de tres modelos. Estos modelos emplean los mismos conjuntos de información que utilizan los tres modelos de regresión logística explicados anteriormente. El primero de los modelos sólo utiliza el texto de la valoración, el segundo modelo concatena el título con el texto de la valoración y el tercer modelo concatena el título, el texto de la valoración y el texto de ventajas/desventajas.

El valor de logprior (probabilidad a priori) para los tres modelos ha sido de 1.47 y la longitud del diccionario de log Likelihood crece con el aumento del diccionario de frecuencias calculado en cada modelo. En la Tabla 5 se muestra el número de claves de cada diccionario.

Modelo 1: valoración	**Modelo 2: título + valoración**	**Modelo 3: título + valoración + ventajas/desventajas**
28.594	30.296	32.045

Tabla 5. Número de claves de los diccionarios de log Likelihood.

7.3.1 VALIDACIÓN SOBRE EL CONJUNTO DE TEST

Para la obtención de las métricas modelo, se ha implementado una función de test que realiza las predicciones del conjunto de test. Los resultados obtenidos se detallan en el Capítulo 8.

7.4 MODELO PRE-ENTRENADO

Un modelo pre-entrenado es un modelo que ha sido entrenado con una gran cantidad de datos para resolver un problema. En el presente trabajo se han utilizado dos modelos pre-entrenados preparados para analizar el sentimiento que hay detrás de un texto.

El uso de estas librerías de referencia del análisis de sentimientos ha permitido comparar los resultados obtenidos con los modelos de clasificación implementados en el proyecto con los resultados que han proporcionado las librerías pre-entrenadas en la clasificación de las valoraciones.

Para poder realizar una comparación de los modelos desarrollados, se han evaluado sobre el conjunto de test los modelos pre-entrenados explicados en el apartado 4.1.3. Utilizando únicamente el conjunto de test para evaluar el rendimiento de estos modelos, se han realizado tres pruebas:

- Clasificación utilizando el texto de las valoraciones.
- Clasificación utilizando el texto de las valoraciones junto con el texto del título.
- Clasificación utilizando el texto de las valoraciones, el texto del título y el texto de las ventajas y las desventajas.

Los resultados obtenidos se describen en el Capítulo 8.

Capítulo 8. ANÁLISIS DE RESULTADOS

En este capítulo se muestran los resultados obtenidos con el desarrollo del proyecto.

8.1 MÓDULO WEB SCRAPING

En cuanto a la implementación del módulo *web scraping*, se ha conseguido obtener una cantidad de valoraciones suficiente para el desarrollo posterior de los modelos de clasificación. El módulo sigue operativo y disponible para la extracción de más valoraciones de cualquier empresa que esté registrada en Indeed. En la Tabla 6 se resumen los resultados de la ejecución del módulo de *web scraping*.

	Valoraciones en inglés	**Valoraciones en español**
Puntuación de 1 a 5	99.484	9.820
Positiva o negativa	78.287	8.170

Tabla 6. Resultados módulo web scraping.

8.2 MODELOS DESARROLLADOS DE NLP

En cuanto a los resultados obtenidos con los modelos de clasificación desarrollados, en el presente apartado se muestran las métricas obtenidas con cada modelo, la matriz de confusión obtenida sobre el conjunto de test, se realiza un análisis del posible error en la predicción de algunas valoraciones y se comparan las métricas obtenidas en entrenamiento con las métricas obtenidas en test.

8.2.1 MÉTRICAS

En el siguiente apartado se van a comparar los resultados obtenidos en cada uno de los modelos desarrollados. La comparación se va a centrar en la mejora que han tenido los

modelos tras incorporado más información con la adición del texto del título de la valoración y con la adición del texto de las ventajas y las desventajas.

A continuación, se detallan las métricas que se han utilizado para la evaluación de los resultados de un modelo. En las fórmulas de las métricas mostradas: *TP* se corresponde con los valores positivos predichos de manera correcta; *TN* se corresponde con los valores negativos predichos de manera correcta; *FP* se corresponde con los valores positivos predichos de manera incorrecta; y *FN* se corresponde con los valores negativos predichos de manera incorrecta.

- *Precision*: es el ratio de las valoraciones predichas de manera correcta y el total de observaciones positivas predichas.

$$precision = \frac{TP}{TP + FP}$$

- *Recall*: mide la relación entre las valoraciones predichas de manera correcta y el total de valoraciones positivas reales.

$$recall = \frac{TP}{TP + FN}$$

- *Accuracy*: mide la relación entre el total de valoraciones correctamente predichas y el total de valoraciones.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$

- *F1 score*: es la media ponderada de la métrica de *precision* y *recall*.

$$f1\ score = \frac{2 \times (recall \times precision)}{recall + precision}$$

En el presente proyecto, se ha considerado la métrica F1Score la base para decidir, puesto que realiza la media ponderada de *precision* y *recall*, mientras que la *accuracy* sólo tiene en cuenta el grado de acierto de una manera genérica.

8.2.1.1 Modelo 1. Información de la valoración.

La Tabla 7 contiene las métricas obtenidas por cada uno de los modelos de clasificación que sólo han utilizado el texto contenido en la valoración.

Modelo	Precision	Recall	Accuracy	F1 score
Regresión logística	0.8311	0.9759	0.8216	0.8977
Naive Bayes	0.9222	0.9183	0.8723	0.9202
VADER	0.8882	0.9167	0.8406	0.9022
TextBlob	0.8635	0.923	0.8217	0.8926

Tabla 7. Métricas modelo con valoraciones.

Los mejores resultados son obtenidos con el modelo de clasificación Naive Bayes. Destaca la baja *precision* del modelo de regresión logística, motivada por el desbalanceo encontrado en el conjunto de entrenamiento (75% valoraciones positivas y 25 % valoraciones negativas). El modelo de Naive Bayes logra un mejor desempeño ya que el parámetro de *log prior* ayuda al modelo cuando las clases están desbalanceadas.

En cuanto a los modelos pre-entrenados, el algoritmo de VADER tiene mejores métricas que el algoritmo de TextBlob sobre el conjunto de test utilizado.

8.2.1.2 Modelo 2. Información de la valoración y título.

En la Tabla 8 se muestran los resultados obtenidos tras añadir el texto del título junto con el texto de la valoración a los modelos de clasificación.

Modelo	Precision	Recall	Accuracy	F1 score
Regresión logística	0.8229	0.9176	0.7986	0.8677
Naive Bayes	0.9299	0.9255	0.8843	0.9277
VADER	0.8966	0.9466	0.8696	0.9209
TextBlob	0.8714	0.9586	0.8533	0.9129

Tabla 8. Métricas modelos con valoraciones y títulos.

Realizando un análisis de los resultados obtenidos con la adición del título, se puede observar que los resultados son mejores, excepto en el modelo de regresión logística. En el último

modelo entrenado se ha introducido el texto de ventajas y desventajas como regresores nuevos dentro del modelo de regresión logística y se ha observado una mejora notable.

El modelo que sigue ofreciendo los mejores resultados sigue siendo el modelo de clasificación de Naive Bayes.

8.2.1.3 Modelo 3. Información de la valoración, título y ventajas-desventajas.

En la siguiente tabla se muestran los resultados obtenidos tras añadir el texto del título junto con el texto de la valoración y con el texto de las ventajas y las desventajas.

Modelo	Precision	Recall	Accuracy	F1 score
Regresión logística	0.8250	0.9871	0.8217	0.8988
Naive Bayes	0.9306	0.9267	0.8858	0.9286
VADER	0.8953	0.9522	0.8723	0.9228
TextBlob	0.8720	0.9628	0.8568	0.9151

Tabla 9. Métricas modelos con valoraciones, títulos, ventajas y desventajas.

Se puede observar una pequeña mejora en el modelo de regresión logística, presentando mejores resultados en la métrica de *precision* respecto a los modelos de regresión logística entrenados anteriormente. Respecto a los otros modelos, se aprecia una pequeña mejora en las métricas, siendo Naive Bayes el mejor modelo de clasificación.

8.2.2 MATRIZ DE CONFUSIÓN

En el presente apartado se muestran las matrices de confusión obtenidas en el conjunto de modelo 3 (que contiene información de la valoración, título y ventajas-desventajas), los que mejores resultados han ofrecido en la clasificación sobre el conjunto de test. La matriz de confusión representa una visualización del desempeño de un modelo entrenado con aprendizaje supervisado. Las columnas representan el número de predicciones de cada clase, mientras que las filas se identifican con la clasificación real. En la Tabla 10 se muestra el ejemplo de una matriz de confusión.

		Predicción	
		Negativos	**Positivos**
Real	**Negativos**	TN	FP
	Positivos	FN	TP

Tabla 10. Ejemplo de matriz de confusión.

8.2.2.1 Regresión logística

		Predicción	
		Negativos	**Positivos**
Real	**Negativos**	464	2.630
	Positivos	161	12.403

Tabla 11. Matriz de confusión modelo 3 regresión logística.

8.2.2.2 Naive Bayes

		Predicción	
		Negativos	**Positivos**
Real	**Negativos**	2.226	868
	Positivos	920	11.644

Tabla 12. Matriz de confusión modelo 3 Naive Bayes.

8.2.2.3 VADER

		Predicción	
		Negativos	**Positivos**
Real	**Negativos**	1.625	1.399
	Positivos	600	11.964

Tabla 13. Matriz de confusión modelo 3 modelo VADER.

8.2.2.4 TextBlob

		Predicción	
		Negativos	**Positivos**
Real	**Negativos**	1.319	1.775
	Positivos	467	12.097

Tabla 14. Matriz de confusión modelo 3 modelo TextBlob.

Analizando los resultados obtenidos, se puede concluir que los modelos que presentan mejor desempeño en la tarea de clasificación atendiendo a la métrica F1 score, ya que también aúna las métricas *recall* y *precision* son los modelos de Naive Bayes. En el apartado 9.2 se describe una serie de tareas a desarrollar en el futuro con el objetivo de mejorar los resultados del resto de modelos.

8.2.3 ANÁLISIS DEL ERROR

En el presente apartado se realiza un análisis de algunas valoraciones mal clasificadas del conjunto de test. Las valoraciones mal clasificadas pertenecen a las pruebas realizadas sobre el modelo 1 de clasificación de Naive Bayes, que incluye sólo el texto de la valoración.

8.2.3.1 Naive Bayes

A continuación, se muestran dos valoraciones que el modelo ha clasificado mal. Se van a mostrar un falso positivo y un falso negativo. La primera es un falso negativo y la segunda un falso positivo.

1. Valoración negativa: "*Its a great place for employees to get along with each other but somethings should not be done in the work place but never the less it was a great job and the staff is very friendly and the welcome you with open arms...*"
   Valoración procesada: ['great', 'place', 'employe', 'get', 'along', 'someth', 'done', 'work', 'place', 'never', 'less', 'great', 'job', 'staff', 'friendli', 'welcom', 'open', 'arm', '...']

2. Valoración positiva: "*It was an okay place to work for good paycheck long hours though upper management didn't really care makes the lower management job harder after 10 years I decided to try throw in the towel*"
   Valoración procesada: ['okay', 'place', 'work', 'good', 'paycheck', 'long', 'hour', 'though', 'upper', 'manag', '”', 'realli', 'care', 'make', 'lower', 'manag', 'job', 'harder', '10', 'year', 'decid', 'tri', 'throw', 'towel']

Tras analizar las valoraciones mal clasificadas, se ha identificado un punto de debilidad de los modelos de clasificación realizados. En muchas ocasiones hay valoraciones negativas que empiezan comentando lo bueno del trabajo, pero después cambian por completo la valoración comentando todo lo malo de la empresa. Este tipo de cambios durante la valoración son imposibles de interpretar por los modelos desarrollados. Otro de los puntos de debilidad del modelo es la no interpretación del léxico de negación, esto impide al modelo cambiar el significado de palabras negativas que tienen una negación delante.

8.2.4 RESULTADOS ENTRENAMIENTO Y TEST

En este apartado se describen los resultados obtenidos en entrenamiento y en test. Se realiza una comparación de ambos y se determina si los modelos presentan *overfitting* (sobreentrenamiento).

El sobreentrenamiento de un modelo ocurre cuando el modelo se aprende los patrones y el ruido de los datos del conjunto de entrenamiento, de tal forma que al recibir datos nuevos no es capaz de interpretarlos de igual manera que los del conjunto de entrenamiento.

Una forma de detectar el sobreentrenamiento es comparando el error sobre el conjunto de training y sobre el conjunto de test. Cuando el error sobre el conjunto de training es notablemente mejor que sobre el conjunto de test puede ser un indicador de que se está produciendo *overfitting*.

En la Tabla 15 se muestran los resultados obtenidos sobre el conjunto de entrenamiento y el de test del modelo 3 de clasificación Naive Bayes.

Modelo	Precision	Recall	Accuracy	F1 score
Naive Bayes test	0.9306	0.9267	0.8858	0.9286
Naive Bayes entrenamiento	0.9391	0.9300	0.8953	0.9345

Tabla 15. Tabla de métricas sobreentrenamiento y test (Naive Bayes).

Como se puede observar en los resultados, no hay sobreentrenamiento. El resto de los modelos desarrollados tienen una relación similar entre las métricas de entrenamiento y test.

Capítulo 9. CONCLUSIONES Y TRABAJOS FUTUROS

En este apartado se describen en líneas generales las conclusiones obtenidas, junto con los posibles trabajos futuros que se han identificado tras la realización del trabajo. Para ello, se analiza el cumplimiento de los objetivos del proyecto y se hace una revisión de las líneas futuras que han surgido tras la realización del presente proyecto.

9.1 CONCLUSIONES

En cuanto al cumplimiento del objetivo principal, que era el **análisis de viabilidad de la aplicación de técnicas de procesamiento del lenguaje natural para el análisis de sentimiento en valoraciones**, se concluye con un grado de satisfacción elevado. El presente trabajo ha realizado un estudio sobre la aplicación de técnicas de procesamiento del lenguaje natural para el análisis de sentimientos, y obteniendo unos resultados adecuados en términos de precisión en las valoraciones extraídas de la página web Indeed. De este modo se puede concluir que la aplicación de las técnicas de NLP es de utilidad para la identificación de sentimientos en las valoraciones extraídas.

En particular, de los resultados obtenidos se deriva que los **modelos de Naive Bayes implementados son adecuados para llevar a cabo la clasificación de valoraciones como positivas o negativas**. Sobre el resto de los modelos, se pueden llevar a cabo tareas para la mejora de su desempeño (descritas en el apartado 9.2).

Los errores analizados al término de la fase de test han permitido detectar los puntos débiles de los modelos. Gracias a esto se han identificado posibles mejoras en los modelos implementados, principalmente en la fase del preprocesado de los textos.

En cuanto a los objetivos específicos, se puede concluir que **se ha conseguido implementar un módulo de *web scraping* que ha permitido el cumplimiento del objetivo principal del**

proyecto. El correcto funcionamiento de esta implementación ha permitido extraer las valoraciones necesarias para desarrollar los modelos de clasificación. Además, **se han conseguido desarrollar dos modelos de clasificación aprovechando el uso de técnicas de procesamiento de lenguaje natural**. Estos modelos han tenido unos resultados muy buenos, mejorando el modelo de Naive Bayes a librerías pre-entrenadas para el análisis de sentimientos. Es importante destacar que el desempeño de estas librerías pre-entrenadas puede mejorar notablemente con las técnicas para lidiar con muestras no balanceadas descritas en el apartado 9.2. Por último, el desarrollo del proyecto ha permitido identificar las aplicaciones del procesamiento del lenguaje natural en contextos realistas, y posibles trabajos futuros a desarrollar.

9.2 TRABAJOS FUTUROS

Tras la realización del presente proyecto, se han identificado trabajos futuros que mejoren los resultados obtenidos y que amplíen las aplicaciones que tiene el procesamiento del lenguaje natural.

En cuanto a desarrollos futuros se han identificado los siguientes:

- Aplicación de técnicas para la mejora de los modelos con muestras no balanceadas: los conjuntos de datos desequilibrados son aquellos en los que la distribución de las clases está muy sesgada y se puede diferenciar una clase minoritaria. Este hecho puede influir en algunos de los modelos de aprendizaje automático, llegando al punto de que el modelo no es capaz de identificar la clase minoritaria. Para abordar este problema existen técnicas que permiten balancear las clases, llamadas submuestreo (*undersampling*), que consiste en eliminar registros de la clase mayoritaria, y sobremuestreo (*oversampling*), que consiste en duplicar los registros de la clase minoritaria. Este tipo de técnicas pueden mejorar el rendimiento de los modelos de regresión logística y los modelos pre-entrenados probados en el proyecto.

- Uso de técnicas de **Deep Learning**: un posible trabajo futuro puede ser el uso de técnicas de *Deep Learning* aplicadas a la clasificación de textos, en el caso del trabajo, de la clasificación de valoraciones positivas o negativas.
- Análisis de la **subjetividad**: existen librerías que permiten la identificación de la subjetividad en los textos (la librería TextBlob).
- **Clustering**: este tipo de técnicas permitiría identificar pequeños grupos en las valoraciones que, junto con un estudio posterior, permite la segmentación de las valoraciones.

En cuanto a las aplicaciones del uso de modelos de clasificación de texto se han identificado los siguientes:

- **Valoración de proveedores**: realizando una extracción de las valoraciones de los empleados de una empresa se puede obtener mucha información importante sobre ella. Además, si se consigue extraer información del futuro proveedor en redes sociales o artículos de prensa, se pueden realizar clasificaciones de estos textos con los modelos ya desarrollados.
- **Campañas de imagen**: este tipo de campañas se realizan en las empresas para construir, reforzar y mejorar la valoración pública de una marca, de un producto o de un servicio. Mediante el uso de técnicas de *web scraping* para obtener las valoraciones que haya de la empresa durante la realización de la campaña, y el uso de modelos de clasificación de las valoraciones extraídas, se puede conseguir medir el impacto conseguido en este tipo de campañas.
- **Sistema de respuesta automática**: uno de los proyectos más ambiciosos y complejos es el desarrollo de un sistema de respuesta automática a las valoraciones recibidas. Actualmente existen algoritmos (GPT-3) catalogados como inteligencia artificial, que interactúan como un ser humano. Son capaces de generar texto, o incluso responder preguntas.

Capítulo 10. BIBLIOGRAFÍA

[1] Ameisen, E. (2019). kdnuggets. Obtenido de https://www.kdnuggets.com/2019/01/solve-90-nlp-problems-step-by-step-guide.html

[2] BOE. (s.f.). Obtenido de https://www.boe.es/doue/2016/119/L00001-00088.pdf

[3] BOE. (s.f.). Obtenido de https://www.boe.es/buscar/act.php?id=BOE-A-1991-628

[4] BOE. (s.f.). Obtenido de https://www.boe.es/buscar/act.php?id=BOE-A-1996-8930

[5] Breitenbach, S. (2020). lengoo. Obtenido de https://www.lengoo.com/blog/milestones-of-machine-translation-part-1/

[6] Brownlee, J. (2020). machinelearningmastery. Retrieved from https://machinelearningmastery.com/random-oversampling-and-undersampling-for-imbalanced-classification/

[7] Christianis, M. (2020). Medium. Retrieved from https://medium.com/analytics-vidhya/scraping-web-apps-using-direct-http-request-f5c02a2874fe

[8] crummy. (2021). Obtenido de https://www.crummy.com/software/BeautifulSoup/bs4/doc/

[9] David Zimbra, A. A. (2018). The State-of-the-Art in Twitter Sentiment Analysis: A Review and Benchmark Evaluation. 29.

[10] excellarate. (2019). excellarate. Retrieved from https://www.excellarate.com/blogs/web-scraping-introduction-applications-and-best-practices/

[11] Git. (2021). Retrieved from https://git-scm.com/

[12] Hanhoon Kang, S. J. (2012). Senti-lexicon and improved Naïve Bayes algorithms for sentiment analysis of restaurant reviews. 11.

[13] ics. (s.f.). Obtenido de https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

[14] ietf. (1999). datatracker. Obtenido de https://datatracker.ietf.org/doc/html/rfc2616

[15] Jaspreet. (2019). Medium. Retrieved from https://towardsdatascience.com/understanding-and-reducing-bias-in-machine-learning-6565e23900ac

[16] Lopamudra Dey, S. C. (2016). Sentiment Analysis of Review Datasets using Naïve Bayes' and K-NN Classifier. 7.

[17] Martin., D. J. (2020). Logistic Regression. Speech and Language Processing., 21. Retrieved from https://web.stanford.edu/~jurafsky/slp3/5.pdf

[18] Monisha Kanakaraj, R. M. (2015). Performance Analysis of Ensemble Methods on Twitter Sentiment Analysis using NLP Techniques. 2.

[19] mulesoft. (2021). Retrieved from https://www.mulesoft.com/resources/api/what-is-an-api

[20] Nadia Felix, E. R. (2014). Tweet Sentiment Analysis with Classifier Ensembles. 31.

[21] Nagappan, M. (s.f.). uwaterloo. Obtenido de https://cs.uwaterloo.ca/~m2nagapp/courses/CS446/1195/Arch_Design_Activity/ClientServer.pdf

[22] nltk. (2021). Retrieved from https://www.nltk.org/

[23] ÖZLÜ, A. (2018). medium. Obtenido de https://ahmetozlu93.medium.com/mastering-rest-architecture-rest-architecture-details-e47ec659f6bc

[24] Peng Shi, W. Y. (2020). Logistic Regression for Sentiment Analysis on Large Scale Social Media Posts via Apache Spark. 6.

[25] Prashant Johri, M. K.-T. (2020). Natural Language Processing: History, Evolution, Application and Future Work. 8.

[26] Raschka, S. (2014). Naive Bayes and Text Classification I. 20.

[27] selenium. (2021). Retrieved from https://selenium-python.readthedocs.io/

[28] Vikas Khullar, A. P. (2017). Sentiment classification on big data using Naïve bayes and logistic regression. 6.

[29] visualstudio. (2021). Obtenido de https://code.visualstudio.com/

[30] webfoundation. (s.f.). Obtenido de https://webfoundation.org/about/vision/history-of-the-web/

[31] wikipedia. (s.f.). Obtenido de https://en.wikipedia.org/wiki/Logistic_regression

[32] Zhang, X. (2018). Forbes. Retrieved from https://www.forbes.com/sites/forbestechcouncil/2018/11/06/the-evolution-of-natural-language-processing-and-its-impact-on-ai/?sh=17a2c7361119

Capítulo 11. ANEXO A

Semana		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	Notas
Comienzo	TAREA	feb 8	feb 15	feb 22	feb 29	mar 7	mar 14	mar 21	mar 28	abr 4	abr 11	abr 18	abr 25	may 2	may 9	may 16	may 23	may 30	jun 6	jun 13	jun 20	
Fase uno	Identificación del proyecto																			FINAL PROYECTO		
	Planificación																					
Fase dos: Módulo Web Scrapping	Exploración páginas web																					Pasos iniciales
	Desarrollo spider																					
	Desarrollo scraper Selenium																					Primera alternativa
	Desarrollo scraper BS4																					Segunda alternativa
	Desarrollo parser																					
Fase tres: Desarrollo modelos de clasificación	Formación NLP																					
	Preprocesado de texto																					
	Análisis exploratorio y sintáctico																					Stemming, Lemmatization...
	Análisis semántico																					NER, WSD, clasificación...
	Sentiment analysis: Regresión logística																					
	Sentiment analysis: Naïve Bayes																					
Fase	Análisis de resultados																					
	Conclusiones y futuros pasos																					

Figura 37. Cronograma de trabajo.