

*Dual Bachelor in Data Science and Telecommunications Technologies Engineering 2020-2025*

***Final Project***

***Machine Learning Applications***



**Alejo González García (100454351)**

**Alonso Madroñal de Mesa (100454449)**

**Daniel Toribio Bruna (100454242)**

**Andrés Navarro Pedregal (100451730)**

# 1.Introduction

This project consists of a multiclass classification of previously preprocessed poems, that will be assigned to a period. In order to carry out this task, we make use of the tools explained in class regarding Natural Language Processing and Machine learning tools, such as feature extraction and selection.

# 2.Task 1: Text Preprocessing and vectorization

## 2.1. The dataset

In order to start the project, we first needed to obtain the dataset to work with. We could either download one available online or create it ourselves. We opted for the second option, and we obtained the information from the web [Poetry Foundation](https://www.poetryfoundation.org/poems/browse#page=1&sort_by=recently_added).

For the creation we used a library called request to obtain the response of the get request and then we obtain the JSON response where we extract information like author, title, link to the poem, tags and snippet. With the link to the poem and using BeautifulSoup we get the html information. Poems were mainly in two formats, text or images. For the first one we just extracted the text that was inside the div with class 'o-poem', and for the second one, we used an OCR, in this case, pytesseract, to extract the text from the image that was in the div with class ‘c-assetStack-media’. TODO ALEJO

In the web [Poetry Foundation](https://www.poetryfoundation.org/poems/browse#page=1&sort_by=recently_added) only 5535 poems from the 47388 total poems have a period assigned, so we will have two different datasets, one composed with all the poems that will be used for the topic modeling task, see Figure 1, and another with only the poems that have a period and they can be used for the classification task, see Figure 2.

The resulting dataset contains the following information:

* An id
* The title of the poem
* The author
* A snippet of the poem
* The link to where the poem is (text or image)
* The categories that the poem has
* The period of the poem
* The text of the poem



Figure 1. Sample of the dataset obtained without periods



Figure 2. Sample of the dataset obtained with periods

## 2.2. Preprocessing

The first thing we are going to do is to remove missing values which means that poem text was missing. This happens because some of the images of the poems were not displayed because of the web. Then we removed the columns that do not give any useful information. These columns are id, snippet and link.

We note during the creation of the datasets that some poems were not in English, but they were translated and published in the web, so we have “duplicated” poems in their original language and in English. To separate the English poems from the others we have used detect from the library langdetect. It is important to know that language detection algorithm is non-deterministic, which means that if you try to run it on a text which is either too short or too ambiguous, you might get different results everytime you run it. To avoid this, we have used a seed. At the end we have obtained that 420 poems are not in English, and as they are translated in the dataset, we can remove them.

After we extract the poems from the web, we saw that some of the characters were not well

encoded, so we fix the characters that are important, for example the apostrophe to be able

to expand the contractions with the method fix from library contractions. Then we tokenize

the text by words, convert the tokens into lower case and filter non alphanumeric tokens. For the homogenization we have chosen lemmatization to keep the semantic meaning and have a better interpretability of the words. Lastly, we have removed the stop words. In addition, we applied N-gram PREGUNTAR SI N-GRAM TIENE SENTIDO CON POEMAS si q tiene. PONER QUE ELIMINAMOS LOS NUMEROS

## 2.3. Text vectorization

Once all the poems are preprocessed and cleaned, the next step is to transform them into a numerical representation that can be used as input for the learning algorithms. We decided to use the following vectorizations:

* Classical BoW or TF-IDF representation.
* Word2vec/Glove based representation or Doc2Vec vectorization. FastText
* Extraction of themes and vector representation of the documents using the LDA algorithm

After obtaining all the different vectorizations, the performance of each one of them with several classification methods will be compared. Then the pair of vectorization and classifier with the best results will be later analysed in order to extract even better results from it.

Prior to beginning with the vectorization step, we shall prepare the dictionary. After analysing the most frequent words in the data set, we can see that there are several words that appear much more times than the others. In order to try to identify these words, we obtained the following figure.

Gráfico, Gráfico de líneas

Descripción generada automáticamente

Figure 3. Most frequent tokens in the initial corpus

We have decided to remove the words that were much more frequent than the others because after applying a filter in the dictionary we saw that these most frequent words are repeated a lot in most of the poems but not in enough to be filtered.

Also, similar graphs can be obtained in order to check the distribution of the words among the different documents of the dictionary. As it has been seen in class, neither the words that appear in all the documents nor the ones that hardly appear in any one are relevant for the algorithm. Thus, it will be useful to have an estimation of this distribution to manually select the discriminant conditions for the removal of both tails of the following histogram.

PONER LAS IMÁGENES DEL NUMERO MEDIO DE TOKENS POR POEMA

BOW and TF-IDF

|  |  |
| --- | --- |
| Gráfico  Descripción generada automáticamente | Gráfico  Descripción generada automáticamente |

## 2.4. Topic modeling