UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DESIGN AND DEVELOPMENT OF A MACHINE LEARNING CLASSIFIER OF ISIS TERRORIST TEXTS

María Alvarez de Sotomayor Vergara
2018

TRABAJO DE FIN DE GRADO

Título:	Diseño y Desarrollo de un Clasificador de Textos Terroristas del ISIS utilizando Machine Learning
Título (inglés):	Design and Development of a Machine Learning Classifier of ISIS Terrorist Texts
Autor:	María Alvarez de Sotomayor Vergara
Tutor:	Óscar Araque
Departamento:	Departamento de Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL CALIFICADOR

Presidente:	—
Vocal:	—
Secretario:	—
Suplente:	—

FECHA DE LECTURA:

CALIFICACIÓN:

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE
INGENIEROS DE TELECOMUNICACIÓN

Departamento de Ingeniería de Sistemas Telemáticos
Grupo de Sistemas Inteligentes

TRABAJO DE FIN DE GRADO

DESIGN AND DEVELOPMENT OF A MACHINE LEARNING CLASSIFIER OF ISIS TERRORIST TEXTS

María Alvarez de Sotomayor Vergara

Julio 2018

Resumen

Hoy en día, el terrorismo es una de las principales preocupaciones de la sociedad. Una de los problemas a los que se enfrentan los organismos que intentan impedir la formación de estas células es la dificultad de controlar un medio de comunicación tan amplio y abierto como Internet. Además, la interacción de las personas en las redes sociales está aumentando exponencialmente en los últimos años, provocando una cantidad incontrolable de datos que sólo pueden ser procesados automáticamente. Por ello, el objetivo principal de este proyecto es la integración de la tecnología como herramienta de detección y clasificación de los textos citados por el ISIS en sus publicaciones en redes sociales.Esto proporcionará un valor importante para acelerar la detección de corrientes de pensamiento grupal, formas futuras de acción o posibles técnicas de reclutamiento utilizadas.

Con el fin de lograr este objetivo,hemos desarrollado un modelo de Machine Learning que ha sido entrenado usando una compilación de 2,685 textos religiosos citados por el ISIS durante un período de 3 años. Además, el modelo ha sido probado en un segundo problema similar: mensajes de ISIS en Twitter con contenido radical y no radical. Para implementar este clasificador se ha desarrollado un sistema software que utiliza técnicas de procesamiento de lenguaje natural (PNL), escrito en Python. En cuanto a la extracción de características, se han estudiado aquellas que proporcionaron información relevante para el modelo considerando nuestra tipología de datos de entrada. Las más destacadas son: PoS,LDA,word embeddings,similitud basada en embeddings y selección de palabras específicas del campo terrorista. Con el fin de evaluar adecuadamente los métodos propuestos, se ha realizado una evaluación exhaustiva basada en validación cruzada.

A modo de conclusión los scorings obtenidos alcanzan un 81% y 92% para los distintos problemas analizados durante el desarrollo del proyecto. Esto se debe a que dichos casos presentaban complejidades diferentes, el primero de ellos disponia de un dataset de pequeñas dimensiones (2250, 2) y cinco posibles tipos de clasificación. Es por ello por lo que algoritmos complejos provocaban la aparición de Overfitting siendo los clasificadores lineares los que mejor se adaptaban a dichos datos. No obstante, el segundo problema era una clasificación binaria y presentaba una base de datos mayor (34708, 2),posibilitando el mejor funcionamiento de algoritmos de alta complejidad.

Palabras clave: NLTK, Scikit-learn, Embeddings, Sentiment, Twitter

Abstract

Nowadays, terrorism is one of society's main concerns. One of the main difficulties encountered by organisms that try to prevent the formation of these cells is the difficulty of controlling such an extensive and open means of communication as Internet. Additionally, interaction of people on social networks is increasing exponentially in recent years, causing an uncontrollable amount of data that only can be processed automatically. Therefore, the main objective of this project is the integration of technology as an enabling tool to detect and classify the texts cited by ISIS in its publications on social networks. This will provide an important value in speeding up the detection of group thought currents, future forms of action or possible recruiting techniques used.

In order to achieve the aforementioned objective we have developed a Machine Learning approach which was trained using a compilation of 2,685 religious texts cited by ISIS over a 3 year period.This dataset is a collection of all the religious and ideological texts used in ISIS English-based magazines. In addition, the model has been tested on a second similar problem: ISIS Twitter posts with radical and non-radical content. To implement this classifier a software system that uses natural language processing techniques (NLP) has been developed, written in Python programming language. Regarding the extraction of features we have studied those that provided relevant information to the model considering our typology of input data. These features are as follows: PoS, LDA, word embeddings, embedding-based similarity, and domain-specific word selection. In order to properly evaluate the proposed methods, an extensive evaluation has been made, based in cross-validation.

As a conclusion, the scorings obtained reached 81% and 92% for the different problems analyzed during the development of the project. This is because these cases presented different complexities, the first of them had a small dataset (2250, 2) and five possible types of classification. This is why complex algorithms caused the appearance of Overfitting, being the linear classifiers the ones that best adapted to these input data. However, the second problem presented a larger database (34708, 2) and a binary classification, enabling better operation of highly complex algorithms.

Palabras clave: NLTK, Scikit-learn, Embeddings, Sentiment, Twitter

Agradecimientos

Gracias a todas las personas que me han acompañado y apoyado durante estos años

Contents

List of Figures

List of Tables

CHAPTER 1

Introduction

1.1 Context

During the nineteenth and twentieth centuries, several elements converged which facilitated and made international terrorism more evident: the creation of smaller weapons with greater destructive power, the publicity generated by a terrorist attack and, above all, technological advances, which allowed for greater speed of movement and communication between terrorists. Mentioning the terrorist group on which the development of this end-of-degree project is focused, we will explain its historical context.

The acronym ISIS [6] refers to the Islamic State of Iraq and Syria but it was in 2006 when it became solely an Islamic State for its members, proclaiming itself as a caliphate. This group claims religious authority over all Muslims, seeking to gain absolute control of most of the Middle East regions inhabited by Muslims. The beginning of this terrorist group's action takes place on September 11, 2001 in the financial and commercial center of New York. The attack on the twin towers by the Al Qaeda group marked the beginning of a new offensive for President George W. Bush, which he called the "War on Terror".

Until April 2013, when the group was officially named ISIS, successive leaders took over, including Osama bin Laden and Saddam Hussein, among others. As of that date, ISIS begins to acquire oil wells and refineries in Syria, from which it will obtain financing to start with crude oil trafficking. Also, the figures of "lone wolves", defined as radical

Muslims acting alone, began to appear.

That is why the alerts began in the western world, leading to intensifications by the US (2014) in the air strikes against the Islamic group. This intensification was joined by the support of 60 countries.

In recent years, the actions of this terrorist group have caused important events such as on 7 January 2015, when two armed terrorists attacked the magazine Charlie Hebdo, or on 13 November of the same year, a series of parallel attacks in Paris killed 133 people. One of the main difficulties encountered by organisms that try to prevent the formation of these cells is the difficulty of controlling such an extensive and open means of communication as Internet.

Additionally, interaction of people on social networks is increasing exponentially in recent years, causing an uncontrollable amount of data that only can be processed automatically. For this reason, it is necessary to incorporate innovative technologies that facilitate the processing and analysis of such data dimensions.

Therefore, the main objective of this project is the integration of technology as an enabling tool to detect and classify the texts cited by ISIS in its publications on social networks. This will provide an important value in speeding up the detection of group thought currents, future forms of action, possible recruiting techniques used or the formation of new cells.

1.2 Project Goals

In order to achieve the aforementioned objective it is necessary to develop a Machine Learning approach which will be trained using a compilation of 2,685 religious texts cited by ISIS over a 3 year period [7]. This dataset is a collection of all the religious and ideological texts (Muslim, Christian, Jewish, and other) used in ISIS English-based magazines.

The main objective is to resolve the problem that causes the classification of the texts posted by ISIS in different websites, developing a text classifier that allows to identify the type of quote mentioned. Among the type of text, we can highlight the following: quotations from the Koran, Hadith or Jihadist.

The main goals of our work are:

- Preprocessing of radical ISIS publications.
- Extraction of relevant features for the subsequent training of the model.
- Study and evaluation of the Machine Learning classifiers that best suit our data typology.

- Building a pipeline that includes all the necessary steps for the subsequent evaluation of results.
- Evaluation of the results obtained with the use of previously identified classifiers.
- Re-training with different feature combinations, modifying the pipeline to increase the final scoring of the model.
- Inclusion of new features like vector representations of words (embeddings) using GloVe, and then extracting the similiraty between them and a Sentiment Dictionary or a Religious Dictionary created from our dataset.
- Testing the model developed on a similar problem: publications in the social network Twitter about radical and non-radical contents. Subsequent evaluation of the results obtained.

1.3 Structure of this document

This document is structured as follows:

Chapter 1: It details the historical context and explains the reasons why this project of design and development of ISIS radical publications classifiers has been carried out.

Chapter 2: This chapter explains the technologies and libraries that have enabled the realization of the project.

Chapter 3: It describes the development of the project as such, from the explanation of the dataset used to the selected classifiers with their respective results.

Chapter 4: The results obtained from the model are presented, making comparisons between classifiers and commenting on the most relevant conclusions.

CHAPTER 2

Enabling Technologies

2.1 Introduction

In this chapter we will explain in detail the technologies that have enabled the development of this project. First, we will highlight Machine Learning, a scientific discipline in the field of Artificial Intelligence that enables the creation of systems that learn automatically. The libraries that have been used will also be detailed, including scikit-learn and pandas. As a short introduction, Pandas provides flexible data structures and Scikit-learn contains all the modules needed for machine learning development (classifiers, regressors, results evaluation...)

2.2 Machine Learning

Machine Learning [14] is a branch of artificial intelligence defined by Arthur Samuel in 1959 as a "Field of study that gives computers the ability to learn without the need to be explicitly programmed". We can classify Machine Learning in three different categories that we explain below:

1. **Regression Method:** The Regression model consists of the prediction of continuous variables. This model is used for other problems than classification.

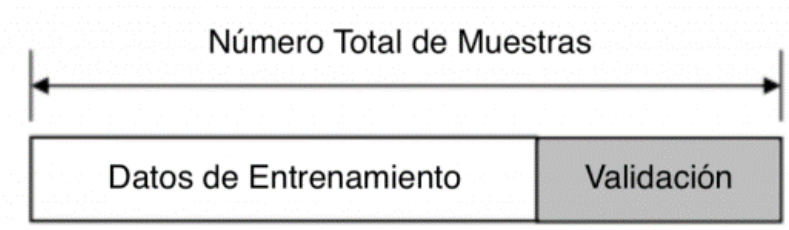
2. **Classification Method:** This method is used to predict the result of an attribute with a discrete value (a,b,c,...) given some characteristics (x_0, x_1, x_2...). There are more or less rigorous methods for classification. The simplest is the binary, which consists of a classification of an input register in 1 or 0. The multiple is an extension of the binary classification.

3. **Clustering Method:** The Clustering method consists in the study of the input data variables for the subsequent construction of coherent groups (cluster) of instances. The algorithms are also classified into two large groups:

   (a) **Supervised Learning**

   This type of algorithm has the special characteristic of having experience or previous knowledge on which to rely to make predictions or decisions. It consists of two phases [4]: the first of training and the second of testing.

   Training: the algorithm learns from a data set (usually 60, 70% of the input data) by finding patterns and structures in the received data. Later in the test phase, a comparison is made with the remaining data (40% or 30%), thus validating the performance of the algorithm.



   Among the most relevant algorithms we can find:

   - Linear Regression
   - Logistic Regression
   - Neural Networks
   - SVMs (Support Vector Machines)

   (b) **Unsupervised Learning** In this type of learning, there is no prior knowledge of the data, so these algorithms are based on looking for structures or patterns of the information they receive. Some of the most important algorithms are the following:

   - K-means

- PCA
- Anomaly Detection

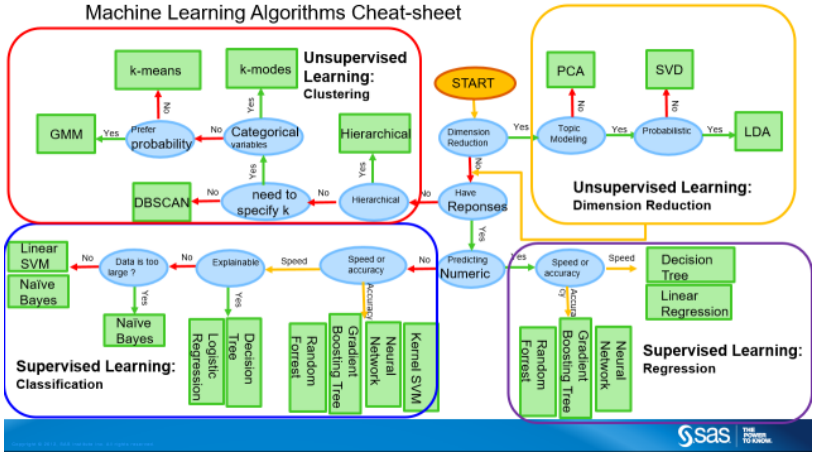


Figure 2.1: Types of Machine Learning and When To Use Them [5]

Depending on the performance of the algorithm and the dataset characteristics, two phenomenons can occur in Machine Learning: *Underfitting* and *Overfitting*.

1. **Underfitting** The selected hypothesis or function inefficiently maps the trend of the training data, causing problems in the generalization of new records. The following changes can be made to fix the sub-adjustment:

   (a) Increase of features that provide relevant information.

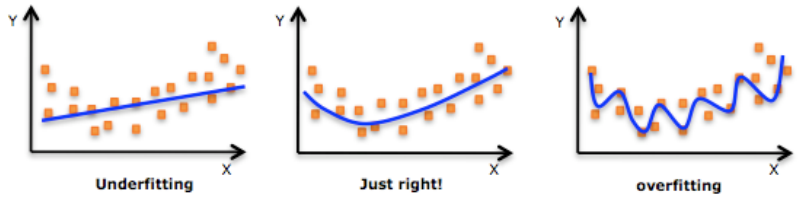
   (b) Increase the number of training instances

2. **Overfitting** The over-adjustment [22] is based on the fact that the algorithm is able to almost perfectly predict the input data, given that it has been overtrained with certain data for which the desired result is known. Therefore, it fails to generalize new registrations. When this happens, the algorithm will only consider as valid the identical data of the training set (including its defects). There are solutions to combat over-adjustment:

   (a) Reduction of number of features

   (b) Select another algorithm that is capable of correctly solve the problem

In the following image the effect of underfitting and overfitting is shown graphically:



Finally, we would like to highlight examples of the use of Machine Learning in daily activities such as the following:

1. **Goo-news:** Classification of contents in web search engines.
2. **Social networking:** Friend referrals, facial recognition, etc.
3. **Email:** SPAM detection
4. **Medicine:** disease predictions, genetic sequencing.
5. **Commerce:** Product customization, Customer segmentation
6. **Netflix, Amazon, iTunes:** Personalized recommendations adjusted to the clients.
7. **Pred-Game:** Predictions of games, such as football, basketball, baseball

2.3 NLTK

NLTK (Natural Language Toolkit) [9] is a Natural Language Processing library that is implemented with the Python programming language. It's an open source library. Provides basic tools for data manipulation. At one extreme, it might be as simple as counting the frequency of occurrence of a word to compare different styles of writing. At the other extreme, NLP involves recognizing and understanding complete human expressions, in order to give you useful answers.

NLTK is available for multiple operating systems and its primary objective is to support research and teaching in NLP including cognitive sciences, information retrieval, machine learning and artificial intelligence. It is being used for individual and teaching use, for prototype research and for construction.

Language processing task	NLTK modules	Functionality
Accessing corpora	nltk.corpus	Standardized interfaces to corpora and lexicons
String processing	nltk.tokenize, nltk.stem	Tokenizers, sentence tokenizers, stemmers
Collocation discovery	nltk.collocations	t-test, chi-squared, point-wise mutual information
Part-of-speech tagging	nltk.tag	n-gram, backoff, Brill, HMM, TnT
Classification	nltk.classify, nltk.cluster	Decision tree, maximum entropy, naive Bayes, EM, k-means
Chunking	nltk.chunk	Regular expression, n-gram, named entity
Parsing	nltk.parse	Chart, feature-based, unification, probabilistic, dependency
Semantic interpretation	nltk.sem, nltk.inference	Lambda calculus, first-order logic, model checking
Evaluation metrics	nltk.metrics	Precision, recall, agreement coefficients
Probability and estimation	nltk.probability	Frequency distributions, smoothed probability distributions
Applications	nltk.app, nltk.chat	Graphical concordancer, parsers, WordNet browser, chatbots

Figure 2.2: Language processing tasks and corresponding NLTK modules with examples of functionality [9]

2.4 GSITK

GSITK is a helper library to perform common operations in NLP, oriented to Sentiment Analysis. GSITK and Senpy are intemately related. The first one is oriented to the process of developing and evaluating a NLP model, while Senpy is oriented to its deployment and use. A NLP model can be constructed and evaluated with gsitk and deployed with senpy. Its main characteristics are:

1. Dataset management
2. Preprocessing utilities (Tokenizer, stop words cleaning, normalizing texts)
3. Feature extraction (WordNet, Word Embeddings similarity, Sentiment/Emotion use of lexicon)
4. Classifiers (Vader Classifier)
5. Evaluation configuration and management
6. Resources management (such as sentiment lexicons)

2.5 Pandas

Pandas is a Python library [12] whose objective is data analysis, providing a flexible and efficient data structure. It was developed by Wes McKinney and is built on NumPy. In

addition,is a free software distributed under the BSD license version three capsules. Python with Pandas is used in a diverse range of sectors including academic and commercial. It offers the following data structure:

- **Series:** One-dimensional indexed arrays, similar to dictionaries.
- **DataFrame:** Data structures of great similarity to the Database tables.
- **Panel, Panel4D, and PanelIND:** Data structures that allow working with more than two dimensions. More complex and less used.

The relevant features of the library as named on its official website are:

- A set of labeled array data structures, the primary of which are Series and DataFrame
- Index objects enabling both simple axis indexing and multi-level / hierarchical axis indexing
- An integrated group by engine for aggregating and transforming data sets
- Date range generation (date range) and custom date offsets enabling the implementation of customized frequencies
- Memory-efficient "sparse" versions of the standard data structures for storing data that is mostly missing or mostly constant (some fixed value)

Pandas solves the disconnect that existed between Python and data analysis modeling, allowing the creation of an entire data analysis workflow in Python without having to switch to a more domain specific language like R.

Furthermore, Pandas does not implement eficient modeling functionality outside of linear and panel regression. For this, we will use scikit-learn.

2.6 Scikit-learn

Scikit-Learn [19] is an open source Machine Learning library for Python. Provides simple tools for data mining and data analysis. It is built on NumPy, SciPy and matplotlib. It provides efficient implementations of state-of-the-art, reusable algorithms for all scientific disciplines and fields of application. It also provides quick and easy prototyping thanks to the leverage of Python's interactivity. It contains the following main functionalities:

1. **Classification:** It is based on identifying the category to which the algorithm entry belongs.

2. **Regression:** predicting an object-related value
3. **Clustering:** grouping of objects with common characteristics
4. **Dimensionality reduction:** Reduction of the variables to consider.
5. **Model selection:** Perform comparisons, parameter and model validations
6. **Preprocessing:** Extraction of relevant features and normalization of texts.

Examples of these algorithms are random forest, grid search, SVM, etc. This library is focused on data modeling. Unlike NumPy and Pandas which focus on the text preprocessing phase (data manipulation and summarizing).

In scikit-learn, all objects and algorithms accept input data in the form of two-dimensional matrices of (number of samples $\times$ number of features). This makes it generic and domain-independent. The objects learned in Scikit share a uniform set of methods that depends on their purpose: estimators can adjust models from data, predictors can make predictions about new data, and transformers convert data from one representation to another.

2.7 SciPy

SciPy [3] is a set of algorithms and mathematical functions built on the NumPy extension of Python. Provides high-level commands and classes for manipulating and displaying data. With Scipy, data processing and system prototyping environments are possible that compete with Matlab, IDL, Octave, R-Lab and SciLab. It is an Open Source library born from Travis Oliphant's original collection. This collection was launched in 1999 under the name Multipack and was made up of extension modules for Python.

One of the main benefits of basing SciPy on Python is to reuse one of the most used and sophisticated programming languages, thus enabling the development of specialized programs and applications. The main modules that SciPy integrates are among others: optimization, linear algebra, integration, ODEs resolution, etc.

2.8 Numpy

Numpy is an [2] essential package for scientific computing with Python. Its origin is based on a module called Numeric that allowed Python to have calculation capabilities similar to softwares like Matlab. Its main mission is to add mathematical and vector capability to Python enabling operation with numeric data or arrays.

2.9 Glove

We define Glove [20] as an unsupervised learning algorithm used to obtain vector representations of words. The result of this model is linear substructures of the Word vector space.

Today, one of the most frequently studied questions is whether distributional word representations are best learned in count-based methods or prediction-based methods. Currently, models based on predictions such as Baroni et al (2014) have an important consideration due to the fact that they are very efficient in a high range of tasks. This model shows that they are not so different from each other as both are able to correctly extract the statistics of the frequency of occurrence of words in the texts. That is why GloVe has been built using the benefit of count data while simultaneously capturing the meaningful linear substructures prevalent in recent log-bilinear prediction-based methods like word2vec.

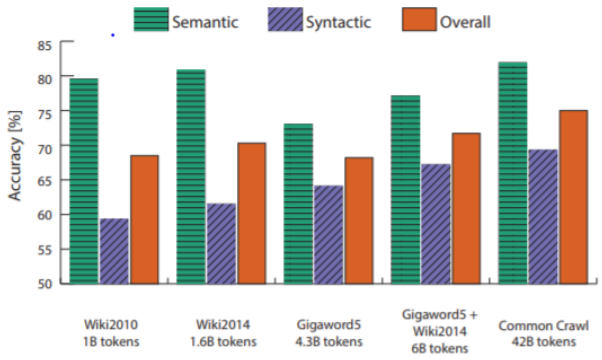


Figure 2.3: Accuracy on the analogy task for 300-dimensional vectors trained on different corpora [20]

As indicated on the Standford website, the training of this model has been done with non-zero entries from a global word-word co-occurrence matrix, indicating how often words occur given other text. But for successive iterations they are much faster since the non-zero matrix entries dimension is much smaller than the initial input data.

As a result, the Global Vector for Word Representation is a log-bilinear regression model for unsupervised learning of word representations which has numerous advantages over previous models such as: Word analogy, Word similarity and Named Entity Recognition tasks.

Compared to Word2Vec [1] which we define as a two-layer neural net that processes text, whose input is a text and its output is a set of feature vectors for each word of the data. Explaining it in more detail, it is based on the search for the transition probabilities

between two states, where each word is identified as a discrete state. This creates vector spaces that contain vectors for similar words.

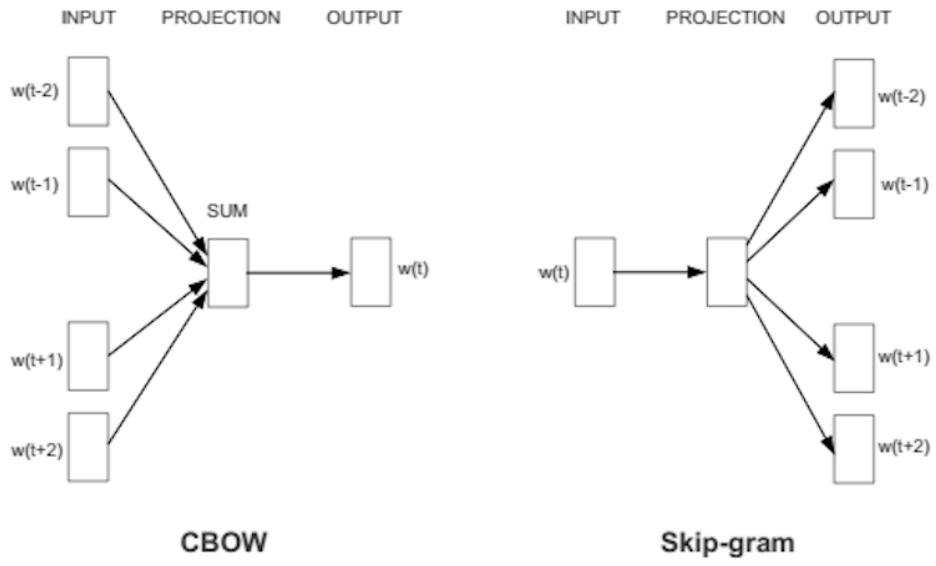
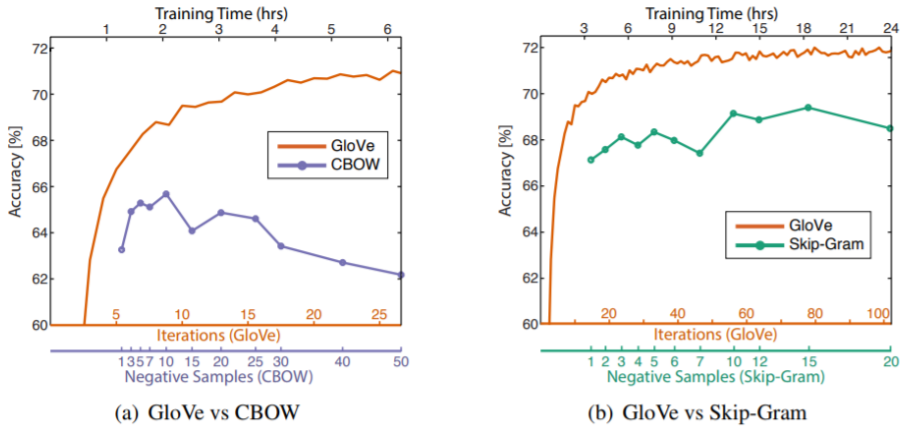


Figure 2.4: Functional diagram of the CBOW and Skip-gram models

In summary, Word2Vec provides numerical representations of word characteristics in vectors. Also, based on past appearances, you can make guesses that allow you to associate words with each other, or to sort documents by topic, etc.



(a) GloVe vs CBOW (b) GloVe vs Skip-Gram

Figure 2.5: GloVe and Word2Vec, both CBOW and Skip-gram comparison [20].

The output of this neural network is a complete vocabulary in which each element has an associated vector, which can then be introduced into a Deep-learning neural network or be consulted to detect similarities between any two words.

To better understand how Word2Vec works we compared it to the operation of an autoencoder, which encodes each word in a vector, training them with others that surround it in the input data. This is done in one of two ways:

1. Using the method known as bag of words or CBOW which relies on context to predict a word

2. Using skip-gram which is based on the use of a word to later predict a context. This method is most effective for large data sets.

Below we can see the results obtained using CBOW vs GloVe and Skip-gram vs Glove (Fig. 2.5).

The figure describes the behaviour on the analogy task as a function of training time. The x-axes shows the number of training iterations for GloVe and negative samples for word2vec. A quite interesting observation is the decrease in performance of word2vec if the negative samples increase by more than 10. This may be because the target probability distribution is not being done correctly.

2.10 Sentiment Dictionary

Sentiment dictionary arose from an investigation [15] that sought to identify the positive or negative opinion after the purchase of a product. The main missions they had were, based on the online opinions collected on a product:

- Identification of the main characteristics of the product on which customers have commented.

- For each of them, identify the positive or negative opinion phrases

- Collect the relevant findings in a summary at the end of the investigation

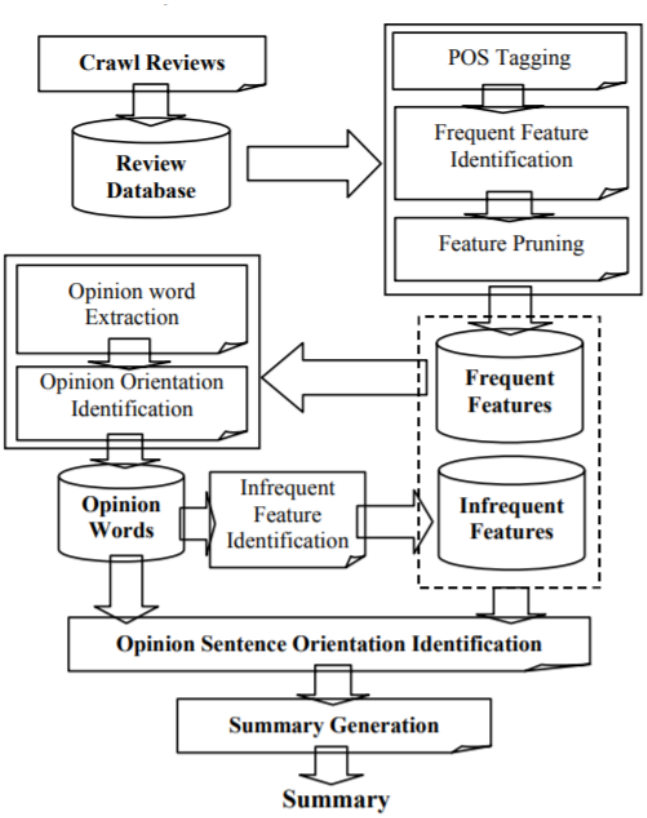


Figure 2.6: Feature-based opinion summarization from [15]

At the start of the process, the system downloads all revisions and stores them in a database. Next, it identifies the most frequent characteristics of opinion. This extracts the words that express these opinions and analyzes their context with WordNet. Considering the mentioned steps, we would like to highlight the relationship with the extraction of the opinion from the customers' comments, which are listed below:

1. **Opinion Words Extraction:** The main problem of these is to be able to distinguish subjective words from those used to describe factual information. Subjectivity has been linked in numerous studies to the presence of adjectives, since they identify an opinion. The extraction of opinion words is also limited to sentences containing one or more characteristics of the product.

2. **Orientation Identification for Opinion Words:** To make a prediction about the semantic orientation of an opinion word, we first need to have identified all the orientations of the data. Semantic orientation is defined as the direction in which your semantic group deviates. We then find three types: 1) Positive Orientation 2) Negative Orientation 3) No Orientation (for example, digital).

For the realisation of this classification a simple method is carried out based on the use of a set of synonyms of adjectives and antonyms of WordNet in order to predict the semantic orientations. As we see in the following figure, adjectives are grouped into bipolar clusters. Behind the head synthesizers, we find the satellites, which group together those similar to the head.

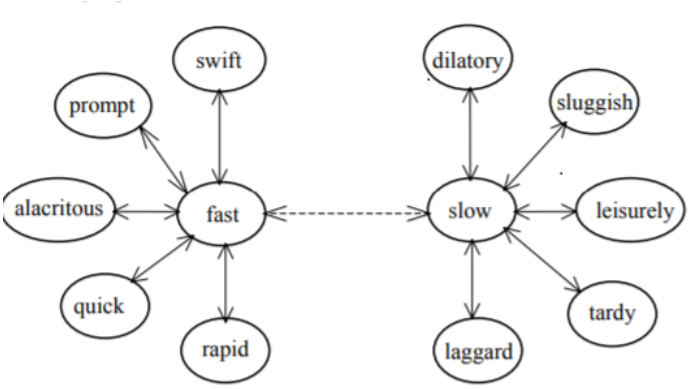


Figure 2.7: Bipolar adjective structure, ($->$: similarity, $-->$: antonymy) [15]

The results of this research were successful as the proposed techniques have resulted in satisfactory experimental results.

CHAPTER 3

Proposed Model

3.1 Introduction

This section will detail the technical development of the project as such, from the acquisition of the dataset and detailed explanation of it, to an explanation of the selected classifiers. Not forgetting the intermediate phases such as the pre-processing that will allow the text to be in the correct format as well as the study and extraction of the relevant features for the subsequent training of the model.

All these steps will also be collected in a pipeline and the feature combinations that have been made to achieve an improvement in the scoring will be explained.

3.2 Data

The dataset used is called "Religious Texts Used By ISIS" [7] and is obtained from the Kaggle platform. The database has 2,685 religious texts cited by ISIS in its publications in English-speaking magazines.

As known, religious texts play a fundamental role in the ideology, propaganda methods and main forms of recruitment of ISIS. For this reason, this dataset gathers all the religious and ideological texts (Muslims, Christians, Jews and others) published in ISIS magazines. This dataset has been created with the main mission that different professional profiles

can reused them in their field of knowledge to understand and propose ideas to counteract radicalisms.

The content is structured in the following columns:

- `Magazine:` Dabiq or Rumiyah
- `Issue`
- `Date` (Month and Year)
- `Type of Text:` The most relevant ones are: Qur'an, Hadeeth, Religious Figure, etc.
- `Source:` What the source of the text was
- `Quote:` The quote itself
- `Purpose:` Support ISIS, Refute Another Group
- The article from which the quote is derived

We show the dataset below, read with the pandas 'read csv' command:

	Magazine	Issue	Date	Type	Source	Quote	Purpose	Article Name
0	Dabiq	1.0	Jun-14	Jihadist	Abu Mus'ab az-Zarqawi	The spark has been lit here in Iraq, and its h...	Support	First Page
1	Dabiq	1.0	Jun-14	Hadith	Sahih Muslim	□The Hour will not be established until the Ro...	Support	Introduction
2	Dabiq	1.0	Jun-14	Jihadist	Abu Mus'ab az-Zarqawi	□The spark has been lit here in Iraq, and its ...	Support	Introduction
3	Dabiq	1.0	Jun-14	Jihadist	Abu Bakr al-Baghdadi	□O Muslims everywhere, glad tidings to you and...	Support	Khilafah Declared
4	Dabiq	1.0	Jun-14	Jihadist	Abu Bakr al-Baghdadi	□O Ummah of Islam, indeed the world today has ...	Support	The World has Divided into Two Camps
5	Dabiq	1.0	Jun-14	Jihadist	Abu Bakr al-Baghdadi	□Therefore, rush O Muslims to your state. Yes,...	Support	A Call to Hijrah
6	Dabiq	1.0	Jun-14	Jihadist	Abu Bakr al-Baghdadi	□We make a special call to the scholars, fuqah...	Support	A Call to all Muslim Doctors, Engineers, Schol...
7	Dabiq	1.0	Jun-	Hadith	Al-Hakim	Abdullah Ibn □Amr narrated that the	Support	The Concept of Imamah

Figure 3.1: Religious Texts Used By ISIS dataframe

From these columns we have selected the following for the realization of our ISIS text classifier:

1. **Type of Text:** this column provides relevant information on the type of text to which the quote belongs. Essential data for the subsequent training of the Machine Learning model.

2. **Quote:** It collects the quotes of the English-speaking magazines previously identified by their typology.

To read the dataset previously explained, we will use Pandas which in short is a Python library containing easy-to-use structures and data analysis. The command we are going to use is "read csv" that will return a Dataframe that we will store in the variable df. A Dataframe is a two-dimensional object tagged with columns of different types. One could say that it has a structure similar to a database table or a spreadsheet. Next, the dropna (inplace=True) Pandas function is used, which would remove the rows where some or all of the data is missing. Once we have stored the content of the Type and Content columns, we count the number of times each type of text (Type) appears and then filter the relevant types such as those that appear more than 100 times. This will cause important changes in the scoring obtained from the rankings, given that as it is a small database, we do not have enough training data for all the Types that appear in that column. The end result of this filtering is the following types:

- **Qur'an:** this being the holy book of Islam containing the word of God revealed to Muhammad by the archangel Gabriel.
- **Hadith:** sayings and actions taken by the prophet Mohammed [21] , written by the people close to him and compiled by the sages who succeeded him.
- **Classical Scholar:** Defined [11] as those publications or statements by religious authorities that are accepted as religious statements by the required communities.
- **Jihadist:** denomination [18] of the most violent and radical branches of political Islam or Islamism, which act to achieve a pretended jihad, which they call holy war in the name of Allah.
- **Taffsir:** is defined as the explanation and critical interpretation of the Koran.

With the changes made, the dataset would consist of the following dimensions: ['Quote', 'Type'] = [2250, 5].

3.3 Preprocessing

After reading the dataset and selecting the relevant columns for the development of the project, we convert the Data Frames into Numpy objects for the execution of the relevant feature extraction step that we will later include as a step in the pipeline.

The pre-processing phase is based on the cleaning of the input text so that it is finally in an ideal condition for the machine to understand it and therefore process it. Text cleansing

is defined as the removal of words, punctuation marks or characters that are irrelevant to the extraction of information and may even cause confusion at later stages.

The first phase is based on Tokenizing the text. That is, divide the text into sentences or words separated by blanks. To do this we use the 'sent tokenize' and 'word tokenize' methods provided by the NLTK library. Once we have stored the list of tokens of all our quotes and types, we will delete the following ones:

- Stop Words: once the tokenized text has been obtained, we proceed to remove the stop words in lower case. In this case NLTK provides us with a list of Stopwords in English that will remove those terms that do not add value for feature extraction. Among them we find: i, me, my, myself, what, that, these, etc.
- Punctuation removal: the punctuation marks contained in the Python String module are removed.

3.4 Feature extraction

Feature extraction is based on applying transformations to input texts in order to extract the most relevant characteristics from them, which then allow texts to be classified more efficiently. In this chapter we will explain exhaustively those used in the project:

1. Lexical Stats
2. PoS Stats
3. LDA (Latent Dirich Allocation)
4. GloVe Embeddings (word embeddings)
5. Embeddings generated from the similarity between a Religious Dictionary and the input data.
6. Embeddings obtained with the similarity of the input data and a Sentiment dictionary.

With this extraction, a pipeline will be built later that will group the steps to follow in order for the training of the classifier and its subsequent evaluation. Several pipelines will be made by increasing the number of features in each version in order to evaluate the improvement of the final score by adding more relevant features.

The feature extractors receive as input the quotes of our dataset previously read and stored in a variable X, and will carry out all the timely operations (pre-processing, operations) to extract the relevant information and add it as one more column in the dataset.

Once the feature extraction is obtained, the FeatureUnion class provided by Scikit-Learn will be used to unify all the data obtained. This class concatenates the results obtained from multiple transformers, having the same purpose as a Pipeline (union and evaluation of parameters). It works as follows: it applies the list of transformers to the input data in parallel and then concatenates the results obtained.

All the implemented transformers have as parameters BaseEstimator and TransformerMixin. The first one inherits the 'getparams' and 'setparams' methods (to adjust the parameters of the estimator), while the second one allows us to fit and transform the data to obtain the required characteristics.

3.4.1 Lexical features Extraction

LexicalStats is defined as the class in charge of extracting the lexical features of each input document. This class has as parameter BaseEstimator and TransformerMixin, whose functionality has been described above.

This transformer implements the 'sent tokenize' method provided by Nltk by subsequently providing a counting of the total number of sentences in the document with the len() function. The finally extracted features are combined in a tuple that contains the total length of the document with the total number of sentences contained in that document.

3.4.2 Word Extraction

To extract words we have used TfidVectorizer [16] which is defined as a converter of a collection of documents to a TF-IDF features matrix. The term TF-IDF (Term Frequency-Inverse document frequency) is a numerical measure that expresses the relevance of a token in a text, being the inverse of its frequency of occurrence. This provides relevant information for obtaining the tokens, which, although less frequent, are a direct classification in their occurrence. It also has numerous adjustable parameters, in our case the following have been set:

- **ngram range:** (1,4). From unigrams to 4-grams, since after trying with numerous combinations this has been the one that has allowed us to obtain better approximations of the context of the quotes of entry.
- **tokenizer:**calling our previously created 'custom tokenizer' method which transforms the input data into tokens.
- **min df:** definition of a float between [0.0, -1.0] that determines the minimum frequency of appearance of a word to be considered in the vocabulary. In our case we

have set this parameter to 0.5, meaning that only words that have a frequency greater than 50.

TF-IDF is the product of two measures, the frequency of occurrence of the term (tf) and the inverse frequency of the document (idf) according to the following formula:

$$w_{x,y} = tf_{x,y} \times \log\left(\frac{N}{df_x}\right)$$

TF-IDF
Term x within document y

$tf_{x,y}$ = frequency of x in y
df_x = number of documents containing x
N = total number of documents

It is equivalent to using CountVectorizer followed by TfidfTransformer.

3.4.3 N-gram Extraction

N-grams extraction is defined as the clustering of n elements of a given text. It is one of the most relevant aspects in feature extraction because it allows the machine to understand a context rather than individual words, increasing the probability of obtaining better results in the classification.

In our case, for example, the word Allah will appear in almost all types of quotes while "by Allah's grace" provides a high probability of being of the Jihadist type. It should be mentioned that CountVectorizer has been used for extraction since it allows to choose the range of n-grams to be taken for extraction.

3.4.4 Part of speech (POS) Extraction

One of the ways to analyze linguistic features is by analyzing grammatical categories. POS Extraction [17] is defined as the assignment of a grammatical category to words in our database. For this we will use a NLTK tagset called 'universal tagset' which is shown in the following table.

Tag	Meaning	English Examples
ADJ	adjective	old, bad, low, special, small, huge
ADP	preposition	on, of, at, with, by, into, under
ADV	adverb	really, yet, still, right now
CONJ	conjunction	and, or, but, if, while, although
DET	determiner, article	the, a, some, most, every, no, which
NOUN	noun	year, home, costs, life, Spain
NUM	numeral	twenty-six, two, 1854, 12:
PRT	particle	at, on, out, over per, that, up, with
PRON	pronoun	he,her,us,its,him,I
VERB	verb	win,said,lost, took
.	punctuation marks	. , ! ?
X	other	gr8,ersatz,esprit

To do this category tagging we have made an extractor that starts with the tokenization of the input text, using NLTK's 'word tokenize' method. Once the tokens are obtained, we tag them in the pos tags previously mentioned in the table. To do this, we have implemented a 'count' variable that obtains the number of pos tag variables that have been obtained for a given quote. Subsequently, a weight classification of a POS category is performed by dividing the number of times that category appears by the total count variable of the quote. In this way, a dictionary will be created whose key will be the POS category and the value of the weight of that category, for each quote.

Finally, the output will be a list of dictionaries of the input dataset, which will later be transformed with DictVectorizer into an actionable format for the machine Learning algorithm that we will see in the following sections. DictVectorizer converts lists from feature-value mappings to vectors.

3.4.5 LatentDirichletAllocation (LDA) extraction

Latent Dirichlet Allocation is defined as a model that generates topics based on the frequency of occurrence of words in texts. This is a great advantage in broadly identifying the issues covered by a document.

This model starts with pre-processing the input text, creating a matrix containing the word count, and then identifying a number of topics according to the previous word specification per topic and total number of topics. The final result consists of a list of topics which in turn contains a list of words associated with the weight corresponding to the relationship with that topic.

3.4.6 Embeddings

This section introduces the proposed model for the analysis of sentiments and religious similarity of the input data.

The following figure shows how the input text is processed by two different submodules:

1. Word/document embeddings

2. Semantic/Religious Similarity

The output of both sub-modules is a vector that is subsequently concatenated and later delivered to a machine learning algorithm, which performs the training based on sentiment and religious annotations.

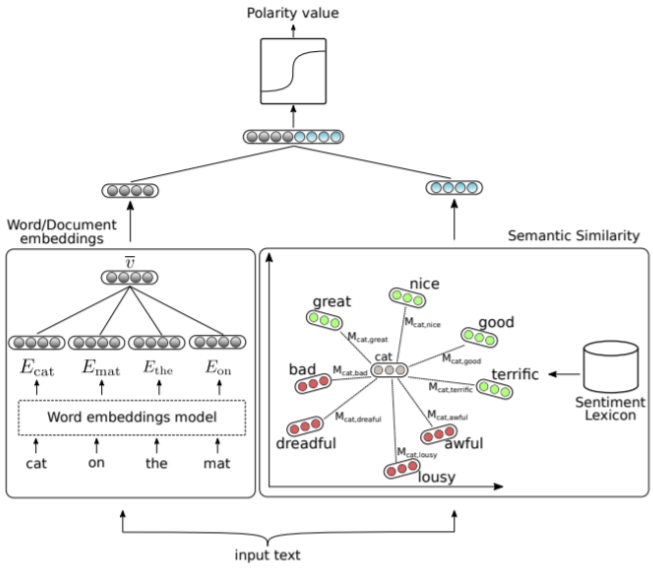


Figure 3.2: Architecture diagram

3.4.6.1 GloVe Embeddings

As we explained in section 2.9 , GloVe is an unsupervised learning algorithm for vector representations of words. Therefore the resulting representation will be a linear substructure of the words in a vector space. In other words, this algorithm gives us a vector for each input word or token. For its later comparison with the Sentiment Dictionary and the Religious one, it has to be in Word2Vec format, since it is accepted by the feature extraction method provided by GSITK called Word2VecFeatures(), for it we use a GloVe to Word2vec converter.

This is due to Word2vec embeddings start with a line with the number of lines and the number of dimensions of the file. This allows gensim to allocate memory accordingly for querying the model. Larger dimensions mean larger memory is held captive. Accordingly, this line has to be inserted into the GloVe Embeddings file.

3.4.6.2 Sentiment feature extractor

To carry out the extraction of sentiment features [8] we will now explain the whole process, starting with the use of a Lexicon Vocabulary and then explaining the similarity function used to obtain the final vectors. These vectors represent the similarity of the input tokens to a series of words chosen from the Sentiment Dictionary.

As we know, the words embeddings contain semantic and syntactic information, however, they do not usually include information about sentiments because they are not included in the training phase of these words. Therefore, we propose the extraction of these features for which the following process is carried out:

We consider the next vector whose length is the same as the input tokens:

$$W^{(i)} = \{w_1^{(i)}, w_2^{(i)}, ..., w_i^{(i)}, ..., w_I^{(i)}\}$$

We also define the set of sentiment words chosen from our Sentiment Dictionary:

$$W^{(s)} = \{w_1^{(s)}, w_2^{(s)}, ..., w_i^{(s)}, ..., w_L^{(s)}\}$$

The dictionary consists of tuples containing the word and its polarity:

$$(w_j^{(s)}, s_j)$$

To carry out the selection of the words in the Sentiment Dictionary, two main criteria are taken into account:

1. Frequency of occurrence of words in training data

2. Grade of information provided by the word obtained from the training annotations

On the other hand, for the feature extraction we perform the following process:

- For each input word $w_i^{(i)} \in W^{(i)}$ and for each sentiment word $w_j^{(s)} \in W^{(s)}$ the similarity function is calculated:

  $$S_{i,j} = \text{sim}(w_i^{(i)}, w_j^{(s)})$$

  where $Si, j \in [0, 1]$, with 0 representing that these words are completely different and 1 that they are the same. After iteration on all the input words and the sentiment ones a $S \in \Re^{I \times L}$ matrix is created.

- A pooling (maximum) function is then applied to the S columns to obtain the vector p of L-length, which contains the semantic similarity feature vector.

  $$pj = maxS_{:,j} = max\text{sim}(w_k^{(i)}, w_j^{(s)})$$

  for k ∈ 1, 2, ..., I

- To calculate the similarity between two words we will use embeddings-based word vector similarity, for which we need a model based on trained embeddings. In our case we use GloVe 2.9. It should also be noted that similarity measurements could have been made using any word embeddings model. The embedding similarity is implemented using the dot product between an input word w_i and a sentiment word l_j :

  $$\text{sim}(w_i^{(i)}, w_j^{(s)}) = E_{w_i^{(i)}}^T E_{w_j^{(s)}}$$

  where $E \in \Re^{|V| \times d}$ is the embedding matrix, and $E_{w_i^{(i)}} \in \Re^d$ and $E_{w_j^{(s)}} \in \Re^d$are the word vectors linked with words w_i and wj.

3.4.6.3 Religious feature extractor

In search of the creation of a feature that provides a notable improvement in the scoring obtained, we designed a religious dictionary that consists of the grouping of the most frequent words in our inbound dataset. Concretely of the quotes mentioned by ISIS, in other words of our input variable X. This dictionary will give us an extra value in feature extraction, since it is built on our input, allowing us to study the features from the inside. To do this we have used Scikit-Learn's CountVectorizer by adjusting the following parameters:

- `ngram_range`: (1,1); this dictionary consist of of unigrams.
- `min_f`: 0.01 which would mean that words with a frequency of less than 10.
- Stop-words: where we indicate that we want to eliminate the stop words of the quotes language, in our case, English.

Next we apply the 'fit transform' method to the data to learn the vocabulary dictionary and return term-document matrix. Afterwards, the most frequent 100 words will be ordered and they constitute the terror dictionary that we evaluate later.

The feature extraction based on the similarity between the Religious dictionary and the embeddings obtained with the `Word2VecFeatures` method provided by GSITK is obtained from the model proposed in the previous section. The only difference found is that instead of using the Sentiment Dictionary 2.10, we use the religious dictionary explained above.

3.4.7 Pipeline

To automate the workflow followed for the implementation of our classifier and its subsequent use, we will implement a Pipeline that groups all the previous steps from the transformation of the input data into a suitable and processable format to the application of them to the Machine Learning algorithms.

To do this, the first step is the phase of pre-processing the input data of our dataset which consists of the tokenization, elimination of stopwords and punctuation marks, as previously explained. Next we move on to the extraction of features, which as we will comment in detail in the following chapter ,has been improved in several pipelines to observe the variation in scoring achieved, causing in some cases overfitting due to the extraction of too many characteristics that resulted in the literal learning of text from the algorithms, causing lower final results.

Below we show the features extracted in the successive pipelines:

1. **Pipeline 1:** LexicalStats(), Words(Tfidf-Vectorizer) , PosStats(), LatentDirichletAllocation()
2. **Pipeline 2:** GloVe Embeddings, PosStats(), LatentDirichletAllocation()
3. **Pipeline 3:** GloVe Embeddings, Glove Embeddings with Sentiment Lexicon
4. **Pipeline 4:** Glove Embeddings, Glove Embeddings with Terror Lexicon
5. **Pipeline 5:** GloVe Embeddings, Glove Embeddings with Sentiment Lexicon, PosStats(), LatentDirichletAllocation()
6. **Pipeline 6:** GloVe Embeddings, Glove Embeddings with Terror Lexicon, PosStats(), LatentDirichletAllocation()
7. **Pipeline 7:** GloVe Embeddings, Glove Embeddings with Terror Lexicon, PosStats(), LatentDirichletAllocation(),LexicalStats(), Words(Tfidf-Vectorizer).

8. **Pipeline 8:** GloVe Embeddings, Glove Embeddings with Sentiment Lexicon, PosStats(), LatentDirichletAllocation(),LexicalStats(), Words(Tfidf-Vectorizer).
   After the feature extraction, the machine Learning algorithm will be implemented and later evaluated.

3.5 Classifiers

After the explanation of the previous phases of the Pipeline (preprocessing and feature extraction), this section will detail the Machine Learning algorithms used in the development of the project. These classifiers will take as input the features extracted in the previous steps and apply the necessary patterns to make the correct classification of the input data into five possible types:

- Qur'an
- Hadith
- Classical Scholar
- Jihadist
- Taffsir

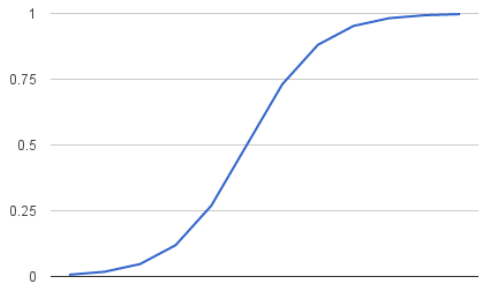
In addition, the following chapter will collect the data resulting from its implementation and will discuss the relevant conclusions.

3.5.1 Logistic Regression

Logistic regression [10], also known as logarithmic regression or maximumentropy classification (MaxEnt), is defined as a linear classification model. In the model, the probabilities that determine the results are performed using a logistics function described below.
The logistic function or also called sigmoid function is a function that was developed by statisticians and that is able to readjust any real-valued number and transform it into a

value between 0 and 1, following the formula below:

$$\frac{1}{(1+e^{-value})}$$



An example of Logistic Regression could be the following:

$$y = \frac{e^{(b_0+b_1*x)}}{(1+e^{(b_0+b_1*x)})}$$

Input values (x) are combined linearly to predict an output value (y). Its main difference with linear regression is that the output is a binary value (0 or 1) and not a numerical one. Detailing the previous formula we observe:

- Y**:** exit
- B_0**:** bias or intercept term
- B_1**:** coefficient for the single input value (x).

Also, all the columns have an associated b-value (constant real number) that has to be learned from the training data. The model values that are relevant are the coefficients of that equation. These are estimated using maximum-likelihood estimation, which is defined as a learning algorithm that makes assumptions about the distribution of the input data.

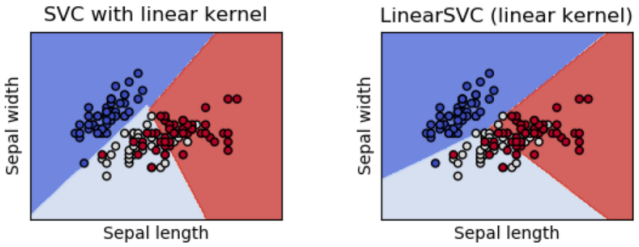
This will provide a model that would predict values very close to 1 for the default class, and 0 for the other class. The logistic regression in scikit learn can be adjusted to binary, rest or multinomial regression with optional regulation.

3.5.2 Linear SVM

Linear SVM is the particularization of the linear kernel of the Support Vector Classification (SVM) model. It has significant improvements such as increased flexibility in the choice

of penalties and loss functions, thanks to the implementation in terms of liblinear instead of libsvm. The input data to the classifier can be sparse or dense. It also provides better results for a large number of samples.

In the following diagram we see that LinearSVM is identical to the use of SVM with the particularization of the kernel parameter = 'linear', but one of the drawbacks is that it loses the implementation of SVM parameters such as 'support'.



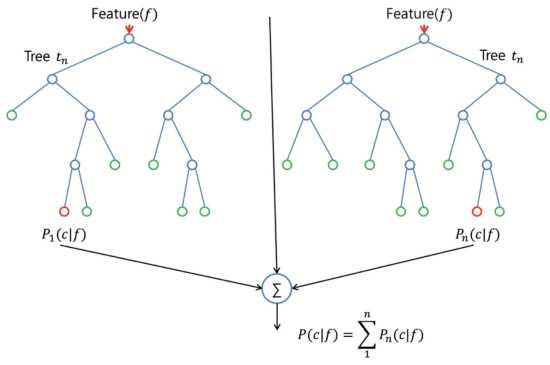
Explaining in detail the objective of SVM is to achieve a minimization of the risk of the equation that follows the model, where the parameters used are C, Loss (L) and Omega (O) that are explained below. The parameter C is used to set the regularization value by telling the classifier how much you want to avoid missclassifying, L expresses the loss function of our examples and model parameters and Omega is like Loss but only penalizing the model parameters.

To correctly set the parameter C we must know that if we choose a high value, the algorithm will try to find the smallest margin hyperplane for the data set. On the contrary, with low values it will separate the values with a hyperplane that will not take into account errors in the dataset classification. In our case, we will use a low value of parameter C, since we have a small dataset.

3.5.3 Random Forest

Random Forest is a meta estimator that improves predictive accuracy and controls over-fitting by adapting to a number of decision tree classifiers in several subsamples of the dataset, whose size matches the original input sample size. If Bootstrap=True the samples are extracted with replacement. It is also a supervised classification algorithm.

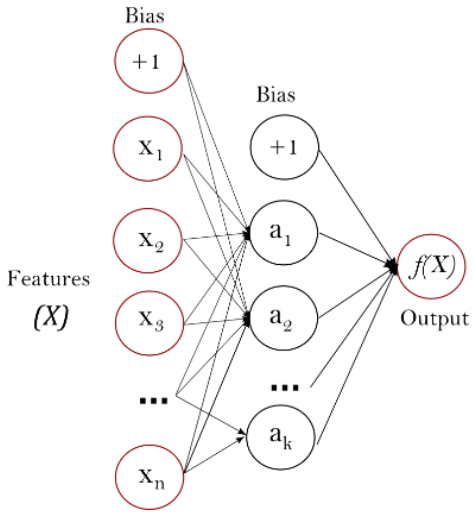
Trees are constructed from a sample of the input data set extracted with replacement. During the construction of a tree, when you divide a node, the chosen division is no longer the best of all characteristics. However, the chosen partition is the best among a random subset of the characteristics causing an increase in the bias of the forest, but causing a decrease in the variance due to the average that results in an improvement of the overall model. The operation of the model is explained graphically below:



As shown in the figure, the model is formed by the implementation of several trees and the classification of a new object from an input vector. The rankings made by each decision tree are represented as votes. As analyzed in the diagram, the first step is the classification of the training set in X subsets for the creation of X decision trees. Afterwards, all of them will vote to which class the object belongs and finally the algorithm will make a total count on the votes, choosing the final class.

3.5.4 Multi-layer Perceptron

Multi-layer Perceptron classifier is a supervised Learning algorithm based on the learning of a non-linear function for regression or classification from input characteristics and a target y. Also, between the input and output there may be one or more non-linear hidden layers as shown in the figure below:



The left part of the diagram shows a series of input neurons that we identify as the different features that later when passing to the next layer will be multiplied by a weight, in

addition to being associated with a non-linear activation function ('relu' or 'tanh'). Finally, the output layer receives the data and transforms it properly to obtain the final values.

The MLPClassifier classifier implements the multi-layer algorithm that is trained using Backpropagation in two arrays: the first one 'X' (quotes in our case) that will have some dimensions (number of samples, number of features) that contains the samples to perform the training, and the second one y that will contain the target values (types in our dataset).

Among the main advantages and disadvantages of this model we find:

Advantages:

1. Ability to learn models in real time using the 'partial fit' function
2. Learning of non-linear models.

Disadvantages:

1. This model requires the adjustment of several complicated parameters to estimate such as the number of hidden neurons and layers, epochs, etc.
2. It presents problems in the scaling of features
3. Different random weight initializations can lead to different validation accuracy.

The activation functions mentioned above are explained below:

3.5.4.1 Rectifier

In the context of Artificial Neural Networks, the rectifier or also known as the ramp function is defined as an activation function in which 'x' constitutes the input of the neuron. This function is analogous to half-wave rectification in electronics. One approach to the rectifier is the analytical function known as softplus:

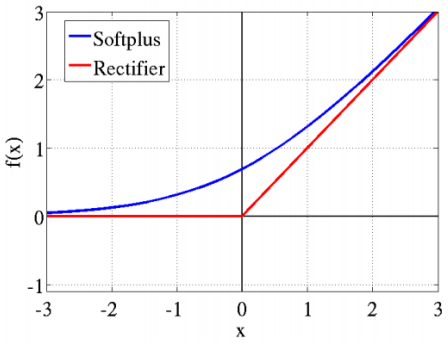


Figure 3.3: Rectifier and Softplus function graph

As shown in the graph 3.3, $f(x)$ is zero when x is less than zero and $f(x)$ is equal to x when it is greater than or equal to zero.

Its main advantages are the following:

1. The function and its derivative are **monotonous**.
2. Its **range** is 0 to infinity.

In contrast, there are the following drawbacks: when the input is negative, the function discards it directly. This decreases the model's ability to fit or train all data correctly.

3.5.4.2 Hyperbolic tangent

This non-linear activation function is similar to sigmoid, mentioned above. Its range is limited between [-1,1] and this gradient is greater than that of the sigmoid function.

The main advantages of its use are grouped below:

1. The function is **differentiable**
2. It is primarily used for **binary classification**.
3. Used in **feed-forward networks**
4. **Negative inputs** will be mapped to negative values and **positive inputs** to values greater than zero, as shown in the graph 3.4.

$$f(x) = tanh(x) = \frac{2}{1+e^{-2x}} - 1$$

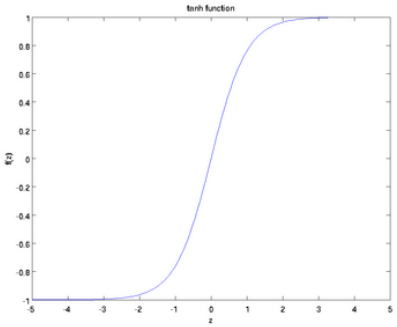


Figure 3.4: Hyperbolic tangent graph

CHAPTER 4

Evaluation

4.1 Introduction

In this chapter we will evaluate the results of the model using cross validation and discuss the most relevant conclusions. For this purpose, 8 pipelines are built consisting of different combinations of the extracted features. In turn, each pipeline will be executed in the selected classifiers: Logistic Regression, Linear SVM, Random Forest and Multi Layer Perceptron, followed by an exhaustive study on the values of the hyper parameters that best fit with the different classifiers.

4.2 Cross Validation

Once all the steps of the Pipeline have been completed, the model created is trained. In our case we are going to use Cross Validation as we have a small dataset (2250, 2).

The cross-validation operation consists of the division of the input data into K subsets and the subsequent training in the K-1 subset elements. This is because the last subset is used for testing. This operation will be repeated K iterations. The following diagram explains this in simple terms:

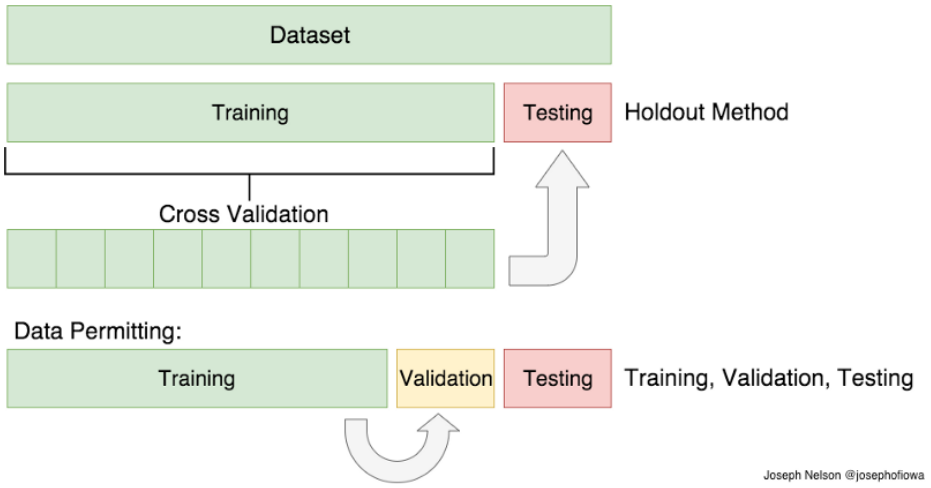


Figure 4.1: Scheme of operation of the cross validation method

Within the cross validation methods , in this project we have chosen K-Fold and GridSearchCV. We start by defining K-fold. This method consists of dividing data into different K subsets or also called folds. As mentioned above, the K-1 subsets will be used for training and the last one for testing. This method will be repeated during K iterations and then an arithmetic average will be calculated with the results obtained in each iteration in order to achieve a precise result. The only disadvantage is that it involves a high computing time.

Once the models that best fit in our input data have been defined, the algorithm's hyper-parameters have to be adjusted to achieve an improvement in the scoring reached. This refinement process is called fine-tuning models. Hyper parameters are defined as those parameters used to describe the configuration of the model in other words, to parameterize the instantiation process of the model. In Scikit-Learn these parameters are passed on as arguments to the estimator builder. In addition, Best Cross Validation Score is used to find the best hyperparameter score. Also, if we want to know the names and values of all the parameters of an estimator, we use the function: 'estimator.get_params()' whose function contains the following parameters:

1. Estimator
2. Parameter Space
3. Method for Sampling Candidates
4. Cross Validation Schema
5. Score function

The combination of parameters can be done in several ways, in our project we have

used GridSearchCV. This method builds and evaluates a model for each combination of parameters defined in a matrix. This procedure consists of the following steps:

1. Comprehensive study of a good combination of hyperparameters.
2. Use GridSearchCV by building a list of hyperparameters to be modified so that all possible combinations are evaluated (using cross validation) and the final result is the parameters that have obtained the highest results in training the model.

4.2.1 Model Evaluation

The main objective of this chapter is to show the results obtained from the different classifiers, as well as from the different combinations of features extracted in the pipelines described below:

Pipeline	Lex	PoS	TF-IDF	LDA	GloVe Emb.	Sentiment Dic.	Religious Dic.
Pipe 1	✓	✓	✓	✓			
Pipe 2	✓	✓	✓	✓	✓		
Pipe 3					✓	✓	
Pipe 4					✓		✓
Pipe 5		✓		✓	✓	✓	
Pipe 6		✓		✓	✓		✓
Pipe 7	✓	✓	✓	✓	✓	✓	
Pipe 8	✓	✓	✓	✓	✓		✓

Table 4.1: Corpus statistics

To do this, the recommended metrics for the classification task are:

1. Precision: performs the proportion of true positives between the total of instances (true and false positives). Its objective is to measure the degree of success of the classifier when it determines that an instance is positive.

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

2. Recall: ratio that determines the accuracy of the classifier to determine all positive samples. It is calculated as the division of true positives into the sum of true positives and false negatives

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

3. F1-Score: average between the precision and recall whose formula is the following:

$$2 \cdot \frac{p \cdot r}{p + r}$$

After analyzing the features extracted in each pipeline and the different evaluation metrics, we proceed to analyze the results after training the model. In the following tables we discuss the results of applying Logistic Regression, Linear SVM and Random Forest classifiers to our Quote and Type input data. Since the F1-score column averages the recall and precision as detailed above, we chose it as the relevant one for the subsequent study of the values obtained. All the results shown in the tables are weighted over a total of 1.

As we can see, the Logistic Regression ranks with a score of 0.78, far surpassing Linear SVM (0.65), and is not far behind RandomForest (0.72).

Class	**Precision**	**Recall**	**F1-score**	**Support**
Classical Scholar	0,72	0,30	0,42	264
Hadith	0,73	0,90	0,81	732
Jihadist	0,92	0,60	0,73	138
Qur'an	0,88	0,98	0,93	1011
Tafsir	1,00	0,08	0,14	105
Avg/Total	0,82	0,81	0,78	2250

Table 4.2: Logistic Regression K-Fold Scores

Class	Precision	Recall	F1-score	Support
Classical Scholar	0,31	0,11	0,17	264
Hadith	0,70	0,67	0,68	732
Jihadist	0,96	0,20	0,33	138
Qur'an	0,74	0,96	0,84	1011
Tafsir	0,24	0,27	0,25	105
Avg/Total	0,67	0,69	0,65	2250

Table 4.3: Linear SVM K-Fold Scores

Class	Precision	Recall	F1-score	Support
Classical Scholar	0,59	0,19	0,29	264
Hadith	0,69	0,83	0,75	732
Jihadist	0,98	0,41	0,58	138
Qur'an	0,81	0,96	0,88	1011
Tafsir	0,93	0,12	0,22	105
Avg/Total	0,76	0,76	0,72	2250

Table 4.4: Random Forest K-Fold Scores

In order to improve the results obtained in the previous section, we will seek to adapt the hyper-parameters of the classifiers. However, we must consider that for some of these hyperparameters, some features extracted may cause the insertion of noise instead of useful information for the classifier.

4.2.2 Model's Classifier Hyperparameter Tunning

To achieve the most appropriate value to be used by a hyperparameter we chose a method provided by Scikit-Learn called GridSearchCV. This consists of perform squares or grid

searches. Grids are formed with all possible combinations of the hyperparameters, scoring each of them against the rest of the hyperparameters. The final result is a $n+1$ dimensional graph with n being the number of hyperparameters and the last one the axis of the punctuation. Cross validation methods will be used for this purpose.

Classifier	**Lexical, Words, PoS, LDA**
LinearSVM	0,69
LogisticRegression	0,78
RandomForest	0,76
MLPClassifier	0,80

Table 4.5: Scores obtained after tunning the classifier's hyperparameters

In the particular case of the classifier that has obtained the best score, MLP Classifier, the search for parameters is slow and thorough, as it is necessary to study beforehand how many hidden layers are considered appropriate. This requires retraining varying the following parameters:

- **Hidden layers:** Number of hidden layers as well as the number of neurons that form those layers.
- **Activation:** Activation function used for these layers. The most commonly used are the so-called rectifier ('relu') and the hyperbolic tangent ('tanh').

After successive tests, the parameters that obtained the best results depending on the features extracted were two 3.5.4:

- **(100,) and 'relu':** A hidden layer of 100 neurons and rectifier as an activation function.
- **(100,100) and 'tanh':** Two hidden layers of 100 neurons each and hyperbolic tangent activation function.

Therefore, in the following tables these two classifiers will be evaluated separately.

Also, looking at the results shown in the tables 4.2, 4.3, 4.4 and 4.6, we can mention the higher scores due to the variation of the hyperparameters made with GridSearch. These variations have been noticeable in all the classifiers as shown in the following table:

Classifier	Score	Variation
LinearSVM	↑	6.15%
Logistic Regression	≡	0%
Random Forest	↑	6,15%

Table 4.6: Score variation after hyperparameter tunning

4.2.3 Cross Validate Evaluation

In this section an exhaustive study will be carried out on the correlation between the extraction of different features and the consequent improvement of the final result. For this purpose, 8 Pipelines with different combinations of features are proposed, as shown in table 4.1. This is due to the fact that in order to search for an increase in the final score, more sophisticated feature extractions are carried out, starting from the syntax or lexical features to the use of embeddings from a religious dictionary created from the most frequent words in the quotes of our dataset or from a feelings dictionary.

The pipelines are divided into two separate tables. The first 4.8 one includes the initial pipeline analyzed in the previous section with the K-Fold and GridSearchCV methods, as well as the rest of the feature combinations including embeddings created from the religious dictionary and GloVe (Gloval Vector for Word Representation). The second 4.7 includes the pipelines containing the embeddings obtained by the Feelings Dictionary (Lexicon) and GloVe. We will carry out an accurate analysis of the results obtained with both dictionaries and subsequently choose one of them.

Discussing the results obtained for the Sentiments dictionary, we observed a maximum score of 79%, obtained by MLP Classifier (100,). At the same time, the situation of Overfitting is repeated as the number of features extracted increases. Likewise, analyzing the rest of the classifiers, we identified 78% of scoring in Logistic Regression and Linear SVM, although with different pipelines.

After making a comparison between pipelines 4, 6 and 8 4.7 with the 3, 5, 7, 4.8 we concluded that the religious dictionary gets a final better score from the model. This was expected as this dictionary is created from the 100 most frequent words of our dataset (not including the stopwords). That is why using this dictionary in later texts, quotes or tweets related to ISIS can significantly improve the probability of success of the developments so far built.

It is worth mentioning that all the results obtained have a probability of success equal to or greater than 75%, which denotes a certain confidence in the implemented classifiers.

Pipeline	Logistic Regression	Linear SVM	Random Forest	MLP (100,)	MLP (100,100)
Pipeline 1	0,78	0,69	0,76	0,80	0,63
Pipeline 2	0,75	0,65	0,75	0,60	0,71
Pipeline 4	0,77	0,70	0,76	0,78	0,77
Pipeline 6	0,78	0,80	0,77	0,80	0,78
Pipeline 8	0,81	0,70	0,78	0,71	0,75

Table 4.7: Results obtained with the first pipeline and it results with the Religious Dictionary

Pipeline	Logistic Regression	Linear SVM	Random Forest	MLP (100,)	MLP (100,100)
Pipeline 3	0,74	0,75	0,74	0,76	0,74
Pipeline 5	0,75	0,78	0,75	0,79	0,76
Pipeline 7	0,78	0,71	0,76	0,68	0,74

Table 4.8: Results obtained with the Sentiment Dictionary

However, the introduction of religious embeddings has not led to an improvement in the MLP(100,0) classifier since the same score (80%) is obtained in both pipeline 1 (PoS, LDA and Syntax features) and pipeline 6, where these embeddings are added. We tend to think that this scoring does not increase in pipeline 8, which links all the feature extraction previously performed, due to the fact that an Overfitting is produced, where the model adapts its structure to the training data set, being unable to generalize in the presence of new data. This also applies to Linear SVM and MLP (100,100). However, this pipeline produces a 1% increase in the highest score obtained in the entire development: achieved by the Logistic Regression ranking with an 81% probability of success.

Pipeline	Lex	PoS	TF-IDF	LDA	GloVe	Sentiment	Religious	Score	Clas.
Pipeline 1	✓	✓	✓	✓				**0,80**	MLP(100,0)
Pipeline 2	✓	✓	✓	✓	✓			**0,75**	LOG.
Pipeline 3					✓	✓		**0,76**	MLP(100,0)
Pipeline 4					✓		✓	**0,78**	MLP(100,0)
Pipeline 5		✓		✓	✓	✓		**0,79**	MLP(100,0)
Pipeline 6		✓		✓	✓		✓	**0,80**	LIN,MLP(100,0)
Pipeline 7	✓	✓	✓	✓	✓	✓		**0,78**	LOG.
Pipeline 8	✓	✓	✓	✓	✓		✓	**0,81**	LOG.

Table 4.9: Best Scores Obtained.LOG:Logistic Regression.LIN:Linear SVM

The highest scores of all the pipelines created are listed above 4.9. Concluding that the classifiers that best fit and work with our dataset are Multi-layer Perceptron Classifier (in the vast majority) and Logistic Regression (obtains the highest score) differing between maximum results by 1%.

CHAPTER 5

Extended Validation

5.1 Introduction

After the evaluation of the problem presented during the implementation of the project, a similar case is presented below where the model will be performed on new inputs. In this way we can observe the effectiveness of standardizing the use of this model for any classification problem related to terrorism. The main differences we find in this problem are the following:

1. The new classifier is binary, due to the fact that the datasets we use are differentiated between radical and non-radical tweets. With this, we aim to achieve an improvement of the previous scoring because the classification problem had five possible cases.

2. The pre-processing phase has been redesigned for the new input data, since the one previously done was for standard publications on websites and magazines while the twitter format contains hashtags and user mentions (@).

3. We carry out an exhaustive study of the pipelines and classifiers that best adapt to the extraction of features and subsequent classification of the data.

Next Section discusses in more detail the used dataset.

5.2 Dataset

The dataset [13] used contains two different sets of data, the first one called Isis Fanboy contains tweets targeting ISIS and the second one About Isis includes a neutral approach to ISIS. This data has been obtained from the Kaggle platform. Speaking a little more in detail about the content of the datasets, we would like to highlight the source of both:

1. **`AboutIsis`** consists of a set of 122,000 tweets collected on two different dates (7/4/2016 and 7/11/2016). It contains terminology related to ISIS such as: Isis, isil, daesh, islamicstate, raqqqa, Mosul, among others. The next caption (Figure 5.1) shows the first ten values of this dataset:

	name	username	tweetid	time	tweets
0	Sean Ferigan	ferigan	752423596182691840	7/11/2016 8:45:39 AM	ISIS influence on the decline as terrorists lo...
1	m.zakariyya	mzakariyya5	752423597172518912	7/11/2016 8:45:39 AM	RT @AyishaBaloch: #IndiaISISandBangladesh And ...
2	ちゃんゆず	yuzuchaaan777	752423592336367616	7/11/2016 8:45:38 AM	@Laika_isis @wink_BF テラリアもってないいいい
3	chutney	plainparatha	752423593548677120	7/11/2016 8:45:38 AM	RT @KabirTaneja: Anti-ISIS volunteer fighting ...
4	ॐ भारत ॐ	dharam_vj	752423588217708544	7/11/2016 8:45:37 AM	RT @MrsGandhi: It 's Urdu dailies not internet...
5	Dipendra Dipzo Khati	DipendraDipzo	752423584757350400	7/11/2016 8:45:36 AM	RT @MrsGandhi: It 's Urdu dailies not internet...
6	James Hasdale	hazzel_j	752423585566724096	7/11/2016 8:45:36 AM	RT @LindaSuhler: " ISIS threatens us today be...
7	Technology Today	Tech42day	752423576955809792	7/11/2016 8:45:34 AM	Islamic State group 's Twitter traffic has plu...
8	MaxxMono	MaxxMono	752423573478834176	7/11/2016 8:45:33 AM	@erasmo83 1) criminalità organizzata si combat...
9	BOSE SUDIPTO	64boses	752423565933228033	7/11/2016 8:45:32 AM	Islamic State group 's Twitter traffic has plu...

Figure 5.1: AboutIsis Dataframe

2. **`IsisFanboy`** has been built on the basis of the publication "How ISIS uses twitter". It includes 17,000 tweets obtained since the attacks in Paris (2015) in more than 100 pro-ISIS accounts. We can see more in detail the columns and the information collected by this dataset in the following figure:

From both datasets we have filtered the "tweets" column as it gives the relevant data for the subsequent feature extraction and model evaluation. To read the datasets we have used the Pandas *read_csv* command, explained in 3. Likewise, the dataset is sized with the same number of tweets of each category (17,354). For this dataframe concatenation, the *concat* command of Pandas is used. To obtain the dataset in a correct format we will add a column that indicates the radicalism of the user, which will have two possible values: "Radical" and "Non radical" allowing a binary classification and the realization of a supervised learning model.

	name	username	tweetid	time	tweets
0	GunsandCoffee	GunsandCoffee70	2.0	1/6/2015 21:07	ENGLISH TRANSLATION: 'A MESSAGE TO THE TRUTHFU...
1	GunsandCoffee	GunsandCoffee70	3.0	1/6/2015 21:27	ENGLISH TRANSLATION: SHEIKH FATIH AL JAWLANI '...
2	GunsandCoffee	GunsandCoffee70	4.0	1/6/2015 21:29	ENGLISH TRANSLATION: FIRST AUDIO MEETING WITH ...
3	GunsandCoffee	GunsandCoffee70	5.0	1/6/2015 21:37	ENGLISH TRANSLATION: SHEIKH NASIR AL WUHAYSHI ...
4	GunsandCoffee	GunsandCoffee70	6.0	1/6/2015 21:45	ENGLISH TRANSLATION: AQAP: 'RESPONSE TO SHEIKH...
5	GunsandCoffee	GunsandCoffee70	7.0	1/6/2015 21:51	THE SECOND CLIP IN A DA'WAH SERIES BY A SOLDIE...
6	GunsandCoffee	GunsandCoffee70	8.0	1/6/2015 22:04	ENGLISH TRANSCRIPT : OH MURABIT! : https://jus...
7	GunsandCoffee	GunsandCoffee70	9.0	1/6/2015 22:06	ENGLISH TRANSLATION: 'A COLLECTION OF THE WORD...
8	GunsandCoffee	GunsandCoffee70	10.0	1/6/2015 22:17	Aslm Please share our new account after the pr...
9	GunsandCoffee	GunsandCoffee70	11.0	1/10/2015 0:05	ENGLISH TRANSLATION: AQAP STATEMENT REGARDING ...

Figure 5.2: IsisFanboy Dataframe

5.3 Preprocessing

The first phase once the input data has been imported is the Preprocessing. As we have explained in the section 3.3, it consists of the elimination of the characters and redundant information for the effectiveness in the construction of the model. In other words, it is based on cleaning the data to obtain a clear and suitable format for the subsequent extraction of features and operation of the chosen classifier.

Previously we had performed this phase for a set of Quotes mentioned in magazines or web publications, where the social network Twitter did not enter. However, the problem now raised includes a number of tweets that involve the use of Hashtags, mentions (@), etc.. For this, we use the GSITK tokenize and preprocess methods which consists of the identification and subsequent tagging of the following elements:

1. **Digits:** Identify the existing numbers in a tweet by adding the `number` tag behind them.

2. **Repeated characters:** In Twitter we usually write informally, so we tend to prolong words, which is an impediment to clearly understand its meaning. Therefore, when this situation occurs, the `repeat` label is added.

3. **Emoticons:** In this social network, we can often give opinions on certain topics or write emoticons to express the degree of agreement or disagreement with a publication, as well as use them to simply reflect a mood on a certain topic. For this reason, a classification is made with the labels `sadface`, `neutralface`, `smile`, `lolface` that depends on the characters used.

4. **URLS:** the urls used in a publication are identified and the `url` tag is added.

5. **Users:** on the social network Twitter, each person is identified by a user, who is recognized in the publications by the @ character. Therefore, once we find this character, we remove it and label the name as `user`.

6. **Capital letters:** We use the String *lower()* method to convert the tweet to lowercase. Previously we identify if a word has been written in its totality with capital letters and we add the label `allcaps`.

7. **Punctuation marks:** Punctuation marks are eliminated as they make it difficult to read and process the information in a correct format.

Once the text is converted to an processable format, we use the method provided by NLTK word tokenize which consists of dividing the document into small parts (numbers, characters, words) called tokens.

5.4 Results

In this section we proceed to re-evaluate the pipelines defined above 4.1 with the different classifiers: Logistic Regression, Linear SVM, Random Forest and Multi Layer Perceptron. With this we will comment on the reliability of the model designed for its later use with different input data.

We will divide the analysis in two tables where we will gather the pipelines that use embeddings obtained from the Religious dictionary, and those that result from the Sentiment dictionary. All results listed below will be decimals with the unit as the maximum possible value (1).

Analyzing the following table we observe that the lowest result obtained is 0.75, which in turn is a high score. We also observe the increase in the score according to the number of features extracted for each classifier (increase in the pipeline number) except for Logistic Regression and Linear SVM, this can be due to numerous reasons, among which we highlight the fact that an Overfitting has taken place. It should also be noted that many pipelines with different classifiers get the maximum result from the model: 0.92. We also identified that the classifiers that best fit the input data in this case are Random Forest and Multi Layer Perceptron.

Pipeline	Logistic Regression	Linear SVM	Random Forest	MLP (100,)	MLP (100,100)
Pipeline 1	0,86	0,80	0,91	0,88	0,89
Pipeline 2	0,88	0,86	0,92	0,91	0,92
Pipeline 4	0,84	0,84	0,91	0,92	0,91
Pipeline 6	0,87	0,87	0,92	0,92	0,92
Pipeline 8	0,81	0,87	0,92	0,91	0,92

Table 5.1: Results obtained with the first pipeline and it results with the Religious Dictionary

Discussing the main differences with the previous table, we observe that there is hardly any difference in the use of the Sentiment or Religious dictionary. Both achieve quite high scorings (0.91 and 0.92) with differences of just decimal places in the different classifiers. However, the previous situation is repeated in the possibility of Overfitting for the Logistic Regression and Linear SVM classifiers. It should also be noted that the Random Forest rankings get the best score in this table.

Pipeline	Logistic Regression	Linear SVM	Random Forest	MLP (100,)	MLP (100,100)
Pipeline 3	0,81	0,89	0,87	0,90	0,87
Pipeline 5	0,85	0,85	0,88	0,91	0,91
Pipeline 7	0,89	0,86	0,92	0,91	0,91

Table 5.2: Results obtained with the Sentiment Dictionary

Next we will group the best results obtained in the following table together with the features that have caused these results:

Pipeline	Lex	PoS	TF-IDF	LDA	GloVe	Sentiment	Religious	Score	Clas.
Pipeline 1	✓	✓	✓	✓				**0,91**	RF
Pipeline 2	✓	✓	✓	✓	✓			**0,92**	RF,MLP(100,100)
Pipeline 3					✓	✓		**0,90**	MLP(100,0)
Pipeline 4					✓		✓	**0,92**	MLP(100,0)
Pipeline 5		✓		✓	✓	✓		**0,91**	MLP
Pipeline 6		✓		✓	✓		✓	**0,92**	RF,MLP
Pipeline 7	✓	✓	✓	✓	✓	✓		**0,92**	RF
Pipeline 8	✓	✓	✓	✓	✓		✓	**0,92**	RF,MLP

Table 5.3: Best Scores Obtained.RF:Random Forest

First, we will mention the main differences found between the results obtained with this binary classification problem and the one developed during this project whose texts had five possible types of classification:

- The lowest score previously obtained was 0.75 compared to 0.90 for this problem.
- The top ranked classifier that achieved the highest score previously was Logistic Regression versus Multi Layer Perceptron and Random Forest in this model.
- There is no presence of the LinearSVM classifier among the highest scores of this model, being one of the most important in the previous problem.
- There is no noticeable difference in the use of the Religious dictionary and the Sentiment dictionary, even though it caused large increases in scores in the previous problem.
- From the use of Global Vector for Word Representation a relevant feature is obtained that causes the highest scoring obtained during the development of the project: 0.92.
- In the first problem analyzed during the development of the project we had a small dataset. In this case we obtained the best score with Logistic Regression, which is a linear classifier. Looking at the results obtained with more complex classifiers we noticed the appearance of Overfitting, this is due to the fact that complex classifiers such as MLP Classifier , work worse with small datasets. However, in the second case, the datasets were bigger, causing better results with complex classifiers (MLP and Random Forest)

CHAPTER 6

Conclusions and Future Work

This chapter contains the main conclusions reached after the implementation of the project (Sect. 6.1), as well as the problems faced and future lines of development (Sect. 6.2).

6.1 Conclusions

This project had the main objective of using Machine Learning techniques to enable the detection and subsequent classification of texts cited by ISIS in its publications on websites, magazines and social networks such as Twitter. Two databases were used for this purpose: the first one was based on the quotes mentioned in magazines and web pages, presenting a classification problem that categorizes text among 5 types: Qur'an, Hadith, Classical Scholar, Jihadist, Taffsir; however, the second one was made up of publications from the social network Twitter, with a binary classification: Radical and Non-Radical. The use of two different input data was due to the pursuit of a standard model that can be scaled for any type of input data.

For the construction of this model we have first designed a different pre-processing phase for each database. This was because their format was different. Twitter uses special characters in its publications such as mentions (@), hashtags (#), etc.

Once the data was obtained in an processable format, we started to analyze the features that gave more information to our model. Eight pipelines with different combinations of features were thus built as described in 4.1.

The most relevant features used in the model have been those obtained from the similarity between the Sentiments/Religious dictionaries and the vector representations of the input tokens obtained with GloVe (Global Vector for Word Representation). The pipelines that implement these feature extractors achieve significantly better results than those that do not.

In addition, after an exhaustive study of the classifiers that best fit the input data, the following were chosen:

1. Logistic Regression
2. SVM with linear kernel
3. Random Forest
4. Multi Layer Perceptron

Cross validation methods (K-Fold) have been used for training and later evaluation. Also, to achieve a notable improvement in the final scoring, a study of the best value of the classifiers' hyper parameters was carried out using GridSearchCV provided by the Scikit-Learn library.

Analyzing the results obtained with each classifier, we divide the analysis into the two different problems raised during the development of the project:

1. **Quotes Classifier:** the most outstanding are Logistic Regression and Multi Layer Perceptron. Both achieve overall results greater than or equal to 0.75, with a maximum score of 0.81.
2. **Twitter binary classifier:** for this problem, the classifiers that get the highest percentages (0.92) are Random Forest and Multi Layer Perceptron.

There are two main reasons that cause the different results between the two datasets:

1. This is because the second problem is simplified. While in the quotes classification we had five possible types that our model had to choose from, the Twitter classifier tackles a binary problem.
2. The datasets used had the following dimensions: (2250, 2) and (34708, 2). We observe that the second dataset is much bigger than the first, which is the reason why it produces better results with complex classifiers such as Random Forest and Multi Layer Perceptron. However, the first dataset, due to its size, falls into Overfitting when used such classifiers. That is why we get the best score with the Logistic Regression, which is a simpler linear classifier.

With this we conclude that all the objectives set for this project have been met, achieving high scores (0.81 and 0.92) for each dataset respectively.

6.2 Future Work

With regard to the future lines of development by which the implementation of this project could continue, we mention the following:

1. Implementation of new feature extractors: In search of an increase in the scoring, an exhaustive study should be carried out on the features that would cause a notable improvement in the learning of the algorithm (NER, Hashtags for the datasets formed by Twitter publications...)in order to improve the developed system performance.

2. Creation of a second more rigorous classifier that provides us with more precise information on the first classification. In this way, our already developed system could be refined by means of a stacked architecture where second-level classifiers analyze the output of previous classifiers, making the annotations more accurate. For example: classification of 'jihadist' quotes could be expanded in order to consider 'attack', 'recruitment', 'doctrine', 'lone wolf', etc.

3. Domain generalization of the project: This would require the creation of a general pre-processing phase that is capable of adapting any type of input data (publications from any social network: Facebook, Instagram...).

Appendix

In order to clarify the identification of impacts of the project developed, we defined a scenario for its analysis. The technology sector of this TFG is telematics systems. With regard to the organisational and strategic scope, this project is part of an international project (Trivalent, H2020-740934), providing the development of software that allows the classification of texts according to their typology. Among the main stakeholders of the project are the bodies in charge of the terrorist investigation: Spanish and international police, CNI, as well as the Intelligent Systems Group (GSI) department, which will incorporate this development as an asset for its subsequent use in the entire project in which it is involved.

Currently, the problem of classifying radical texts is done manually, i.e., a person is responsible for identifying the type of text that is being published as terrorist content on a social network, magazine or website. That is why relying on technology will be a clear benefit replacing repetitive tasks because of its low final value. It will also improve efficiency by significantly reducing the execution time of these tasks. In contrast, a small group of people could be harmed by having their jobs replaced. However, with reference to the ethical aspects, technology should never be a substitute for the person in the meticulous and intelligent work. Nevertheless, in repetitive and boring processes such as the one we are currently dealing with, this could mean a significant improvement that could be used by the worker to speed up part of his work, contributing its value in the parts of the project that conclude with more value.

Ethical, social and environmental impacts and problems related to projects are discussed below. These impacts are summarized in the following table:

Life Cycle	Ethical and professional aspects	Economic aspects	Social aspects	Environmental aspects
Raw materials	Environmental harm	Integration and social acceptance of innovation	Safety and risk prevention. Non-renewable materials	Electrical energy for computer operation (high consumption and efficiency; source: power grid, radiation, vibration, heat radiation)
Design, production and testing	Supportive of culture and cultural diversity	Business development. Productivity improve-ments.Capacity building (workers, users,...)	Access to Information Education Informed Consent and Intellectual Property	x
Use and maintenance	Just distribution of primary goods, capabilities, risks and hazard	Revenue improvement	Right to privacy, data protec-tion.Respect for labour rights, in the company itself and in the supply networks	x
Reuse and disposal	Social inclusion	Integration and social acceptance of innovation	Citizen participation	Waste (hazardousness of waste; difficulty of disposal)

The main impacts are described, detailing the stakeholders concerned, the regulations, laws and ethical codes related to them, as well as the possibilities for evaluation and quantitative assessment:

- Impact 1

    1. **Description**:The development of this Project would imply an Improvement in Productivity, since it would eliminate the slow classification of texts that have to be analyzed individually. This impact would be reflected within the economic ones.
    2. **Groups/sectors concerned**:The sector affected would be the group of workers belonging to a company that is responsible for carrying out this task, positively affecting them by being able to contribute value to more important tasks.
    3. **Regulations, laws, standards, ethical codes of reference** :There are no regulations, laws or restrictions applicable to this impact.
    4. **Evaluation possibilities**:Today, technology is pioneering process automation using advanced robotics (RPA: Robotic Process Automation).Companies from all sectors pursue agility, efficiency and speed in their processes.
    5. **Economic implications**:This impact would bring an economic benefit as it would save considerable time and would permit time investment in other work, avoiding overtime charges for workers or the total reduction in the hours dedicated to a project (savings in personnel costs).

- Impact 2

    1. **Description**:Environmental impact caused by the materials used for the development of the project: Computers and Servers.
    2. **Groups/sectors concerned**:This impact has a negative impact on the environmental sector resulting from the use of this resource.
    3. **Regulations, laws, standards, ethical codes of reference**:For the development of this project, the National Environmental Impact Assessment System (SEIA) will have to carry out a study on the impacts it could have. It is a unique and coordinated system for the identification, prevention, monitoring, control and early correction of negative environmental impacts resulting from human actions expressed through the investment project.
    4. **Evaluation possibilities**:The European Commission collaborates with multinationals analysing the impact of new technologies in order to reduce their environmental impact.

5. **Economic implications**:This impact has no economic implications.

- Impact 3

1. **Description**:For the development of this project, it is necessary to consider Access to information that is identified as a social impact.
2. **Groups/sectors concerned**:This impact will affect the privacy of users of social networks, websites, blogs, etc., as the incoming databases will be built from these publications. This, in turn, does not have a negative impact, since the user has previously accepted the permission to use the content.
3. **Regulations, laws, standards, ethical codes of reference**:Based on the new General Data Protection Regulation (GDPR), the datasets used will have to be anonymous, so the user mentions in the tweets must be eliminated.
4. **Evaluation possibilities**:All companies have updated their privacy conditions given the revolution caused by the leakage of data from Facebook to Cambridge Analytica. Therefore, access to information and user privacy are issues covered.
5. **Economic implications**:This impact may lead to an economic implication if the sources from which the information is obtained are private, having a cost per acquisition of the required information.

- Impact 4

1. **Description**:The construction of this project will have a positive impact on Business Development as well as the integration and social acceptance of innovation. These impacts are within technological sustainability and internal socio-economic aspects.
2. **Groups/sectors concerned**:The affected group/sector will be the employees who have to learn to use this new technology and software program. For this, they will need training sessions in case they do not know how this technology works.
3. **Regulations, laws, standards, ethical codes of reference**:There are no laws or restrictions that apply to this impact.
4. **Evaluation possibilities**:Today we are at the peak of the technological age, this means that all companies are integrating innovative technologies to replace traditional and repetitive jobs.
5. **Economic implications**:The economic impact would be the cost of a teacher or professional in the field of Artificial Intelligence and Machine Learning to give

a training session on its use. Likewise, this would cause an economic growth of the company caused by the saving of qualified personnel for the accomplishment of repetitive tasks, being able to contribute value in meticulous and more sophisticated tasks.

All the aspects listed above have been considered when developing the TFG, so that negative (environmental) impacts can be minimised and positive ones (Impacts 1,3,5) enhanced.

The main ethical dilemma related to the project will be discussed in depth below. As we know, the development of this TFG has been based on the construction and design of a classification of ISIS publications in social networks, websites and magazines. If this classifier is used in a police situation and involves decisions based on the outcome of our system, a failure could lead to major dilemmas given the degree of responsibility that a determination to identify a person as radical would imply without it being so.

Appendix

The development of the project involves the following estimated costs:

1. **Material**

   - **Tangible** Personal computer with i7 processor valued at 1000 €. 12-core server: The GSI HUB server has been used for free, we estimate the price of such a server in Azure:
     - A8v2 Instance
     - 8 cores
     - 16,00 GB ram
     - 80 GB temporary storage
     - 0.506 €/hour

     The total number of computing hours is estimated at 100 hours.
   - **Intangible** No user licenses are required since using Jupyter Notebook which is defined as an open-source web application, its use would be completely free.

2. **Salary** Considering a user with no previous experience in the sector, it is estimated that 10€/hour is needed. The total number of hours dedicated to the project (technical development plus report writing) amounts to 330h, so the net salary of the worker would be 3300 €.

3. **Taxes**

   Taking into account the general data and the employment situation of the worker (23 years old, Technical Engineer, without children) the fees would be as follows:

   Gross annual salary
   3.300 €
   Personal income tax (IRPF) payable 18%
   594 €
   Annual net salary
   2.706,5 €

Bibliography

[1] *Deep learning with word2vec*. https://radimrehurek.com/gensim/models/word2vec.html.

[2] *NumPy v01.14 Manual*. https://docs.scipy.org/doc/numpy/reference/.

[3] *Scipy Tutorial*. https://docs.scipy.org/doc/scipy/reference/tutorial/general.html.

[4] Aprende machine learning, Diciembre 2017. https://blogs.deusto.es/bigdata/tag/underfitting/.

[5] Types of machine learning algorithms and when to use them. *What's The Big Data?*, 2017.

[6] Isis:origen, objetivos, financiamiento y líderes del estado islámico de irak y siria. *Historia y Biografías*, https://historiaybiografias.com/isis/.

[7] Khuram Andrew Thompson. Religious texts used by isis, September 2017. https://www.kaggle.com/fifthtribe/isis-religious-texts.

[8] O. Araque, G. Zhu, and C.A. Iglesias. A semantic similarity-based perspective of affect lexicons for sentiment analysis. Technical report, Universidad Politecnica de Madrid, Avenida Complutense 30, Madrid, Spain, 2018.

[9] Steven Bird, Ewan Klein, and Edward Loper. Natural language processing with python. 2009.

[10] Jason Brownlee. Logistic regression for machine learning. *Machine Learning Mastery*, 2016. https://machinelearningmastery.com/logistic-regression-for-machine-learning/.

[11] caliphate online. Which classical scholar said the aqeeda of islam is spiritual and political? *Caliphate Online*, 2017.

[12] Daniel Y. Chen. *Pandas for Everyone: Python Data Analysis*. Pearson Addison-Wesley Data and Analytics, December 2016. `https://www.tutorialspoint.com/python_pandas/index.htm`.

[13] Sander Dsangeetha, ActiveGalaxy. Tweets targeting isis, September 2016. `https://www.kaggle.com/activegalaxy/isis-related-tweets/data`.

[14] Arquitecto de Soluciones Fabián A. Contreras. *Introducción a Machine Learning*. Sunqu, 2016. `https://www.zemsania.com/recursos-zemsania/whitepapers/DTS/Machine_learning.pdf`.

[15] Minqing Hu and Bing Liu. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177. ACM, 2004.

[16] Schütze H. Manning C. D., Raghavan P. *An Introduction to Information Retrieval*. Cambridge University Press, 2009.

[17] Jamal R. Nassar. *Natural Language Processing with Python*. Chapter 5.

[18] Jamal R. Nassar. *Globalization and Terrorism: The Migration of Dreams and Nightmares*. p.87, 2005.

[19] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

[20] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014.

[21] Dr. Abu Aminah Bilal Philips. *Estudios Islámicos Libro 1*. 2016.

[22] Alejandro Rayón. El machine learning en la era del big data, Marzo 2016. https://blogs.deusto.es/bigdata/tag/underfitting/.