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 AN AGE CLASSIFICATION SYSTEM OF TWITTER USERS BASED ON MACHINE LEARNING TECHNIQUES

ALBERTO MARQUÉS ALBA

2017

TRABAJO FIN DE GRADO

Título:	Diseño y desarrollo de un sistema clasificador de usuarios de Twitter basado en técnicas de aprendizaje automático
Título (inglés):	Design and development of an age classification system of Twitter users based on machine learning techniques
Autor:	Alberto Marqués Alba
Tutor:	Carlos A. Iglesias Fernández
Departamento:	Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL CALIFICADOR

Presidente:

Vocal:

Secretario:

Suplente:

FECHA DE LECTURA:

CALIFICACIÓN:

Resumen

Hoy en día, la forma en que la gente interactúa ha cambiado gracias a la gran revolución que ha supuesto el uso de internet, alcanzando prácticamente cualquier rincón del planeta. Casi todo el mundo está conectado, por lo que la gente puede acceder a la información y compartirla fácilmente con sus dispositivos en cualquier momento.

Uno de los mayores cambios sociales que el mundo ha sufrido en los últimos años es la aparición de las redes sociales, donde se comparte información de todo tipo generando una inmensa cantidad de datos. Toda esta información incluye pistas que pueden ser usadas para elaborar un perfil con atributos del usuario como el género, edad, gustos, religión, etc.

La demanda de este tipo de información está aumentando principalmente debido a su alto valor comercial que puede ser usado por las empresas, por ejemplo, para la elaboración de una campaña comercial adaptada a sus clientes. En este proyecto se analizará este tipo de cualidades centrándose en la edad y la fuente de información escogida es Twitter.

Este análisis se basa en las diferencias de comportamiento observadas en las redes sociales entre los diferentes grupos de edad, que se hacen evidentes en el uso del lenguaje con diferentes expresiones y estructuras gramáticas.

La principal tarea es el desarrollo de un sistema clasificador que permita predecir la edad de un usuario de Twitter. Para ello, el lenguaje de programación escogido es Python con la ayuda de herramientas de aprendizaje automático como Scikit-learn.

Una vez analizados los requisitos del proyecto se ha procedido con la implementación del sistema utilizando como fuente de análisis un conjunto de datos compuesto por Tweets organizados por usuarios en diferentes idiomas.

Para la evaluación del sistema hemos utilizado distintos modelos, comparando sus resultados para escoger aquel con un mejor desempeño.

Finalmente se ha implementado el sistema como un servicio gracias a la creación de un plugin en la plataforma Senpy.

Palabras clave: Aprendizaje Automático, Python, Scikit-learn, Edad, Twitter

Abstract

Nowadays, the way in which people interact has changed thanks to the great revolution that the use of Internet has supposed, reaching almost every corner of the planet. Almost everyone is connected, so people can access and share information easily with their devices in every moment.

One of the greatest social changes that the world has suffered in the last years is the appearance of the social networks, where people share information generating a huge quantity of data. All this information hides clues that can be used to elaborate a profile with attributes of the user like gender, age, likes, religion, etc.

The demand of this kind of information is increasing mainly due to its high commercial value that can be used by the companies, for example, for the elaboration of a commercial campaign adapted to its clients. In this project we are going to analyse this kind of attributes focusing on the age and the data source chosen is Twitter.

This analysis is based on the behaviour differences that can be observed in the social networks between the different age groups, that are evident in the use of the language with different expressions and grammatical structures.

The main task is the development of a classification system that enables predicting the age of a Twitter user. For this task, the chosen programming language is Python with the help of machine learning tools as Scikit-learn.

Once the requirements of the project have been analysed, we started with the implementation of the system as analysis source a set of data composed by Tweets organized by users in different languages.

For the evaluation of the system we have used several models, comparing their results to choose the one with a better performance.

Finally the system has been implemented as a service thanks to the creation of a plugin in the platform Senpy.

Keywords: Machine Learning, Python, Scikit-learn, Age, Twitter

Agradecimientos

Me gustaría dar las gracias a todas aquellas personas que han influido de manera positiva en mí.

A mi familia, sin cuya ayuda y apoyo no habría podido llegar hasta aquí.

A mis amigos y todas esas personas que he conocido en esta etapa, que han hecho que los momentos difíciles sean más llevaderos y que los buenos momentos sean inolvidables.

A mi tutor, Carlos Ángel Iglesias, por guiarme en el desarrollo de este proyecto.

Gracias a todos.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Context

Nowadays, thanks to the internet connection, the use of social networks has been generalized and most people interact through any of the different alternatives available [2].

One of the most popular social networks is Twitter where people share information in form of short text messages of 140 characters. This continuous exchange of information is generating a huge amount of data that can be analysed with different purposes.

In this project we are focusing this analysis as an author profiling problem trying to identify a profiling aspect, more specifically trying to discern the age group of the author of the text messages, called tweets, between five predefined groups.

This is an interesting task because this kind of information is not stored or shared by Twitter so the data obtained can be really useful and could have lots of applications like the elaboration of specific contents for a determined profile of users or its use in advertising purposes.

The development of this task is based on the observation of behaviour differences in the social networks between the different age groups, like the use of different grammatical

structures or writing styles.

These profile aspects, in most of the cases, could not be discerned by a human just reading and interpreting the tweets so we are going to face the problem with the help of machine learning techniques and the use of hidden information that could hardly be detected by a human.

1.2 Project goals

Sometimes the amount of data that we have to analyse can be really hard to manage making it a difficult task. We could say that the main goal of this project is to show the goodness of the use of machine learning techniques to obtain valuable information from the analysis of a big amount of data and to show how to organise and structure all the data in order to be able to obtain this information.

This project is based on the use of classification techniques trying to identify the belonging of an instance between different class groups. More specifically we are going to try to predict the age of Twitter users between some predefined age groups. The objective is to perform a system with the best accuracy possible analysing which features are more relevant in this process so we can improve the development. It is also very important to take into account the resources used and analyse if the results are worth it.

1.3 Methodology

The objectives fixed for the development of this project are translated into a set of tasks to be done that are listed below:

- Organization and preprocessing of the data provided in the dataset of tweets so we can manage it.
- Implementation of the age classification system following the next steps:
  - Definition and extraction of the different features that are going to be analysed by the system.
  - Separation of the preprocessed data into train and test sets in order to use it in the development of the classification system.
  - Election and testing of different algorithms so we can obtain the first results and compare them.

- Use of different datasets and research of new feature extraction techniques so we can improve the performance and efficiency of the system.
- Development of a Senpy plugin where our system will be deployed so we can use it as a real application making real-time predictions.

1.4 Document structure

In this section we provide a brief overview of the chapters included in this document after this introduction chapter. The structure is the following:

The next chapter is the Enabling Technologies chapter (***Chapter 2***), where the main technologies used for the development of the project are explained.

After this chapter we can find the architecture of the project (***Chapter 3***). Here we provide an overview of the set of modules that form the project and how they interact.

On ***Chapter 4***, we explain the different tests performed, the results obtained and how to interpret them.

On the last chapter (***Chapter 5***) we find the conclusions and future work chapter as closure of the project.

CHAPTER 2

Enabling Technologies

There is one main technology arround which the project is developed, Machine Learning. It is a subfield of computer science that comes from the evolution of the study of pattern recognition techniques and computational learning theory in artificial intelligence. Depending on the characteristics of the learning system, it can be classified into supervised learning, unsupervised learning and reinforcement learning. In this project we are going to use supervised learning techniques applied on classification tasks.

For this kind of techniques it is necessary a labeled training dataset used to analyse its characteristics mapping new examples. Labeled training data is formed by a list of examples with an input object, typically a vector, and its output value. The classification inputs are divided into different classes, which are assigned by the learning model to each of the inputs. Classification is usually strongly related to supervised learning due to classification problems contain input objects with their classes as output values.

It is necessary to emphasize the fact that it is necessary to train the classifier with a collection of labeled examples making possible to look for patterns and learn them in order to use them in the classification of new unlabeled samples.

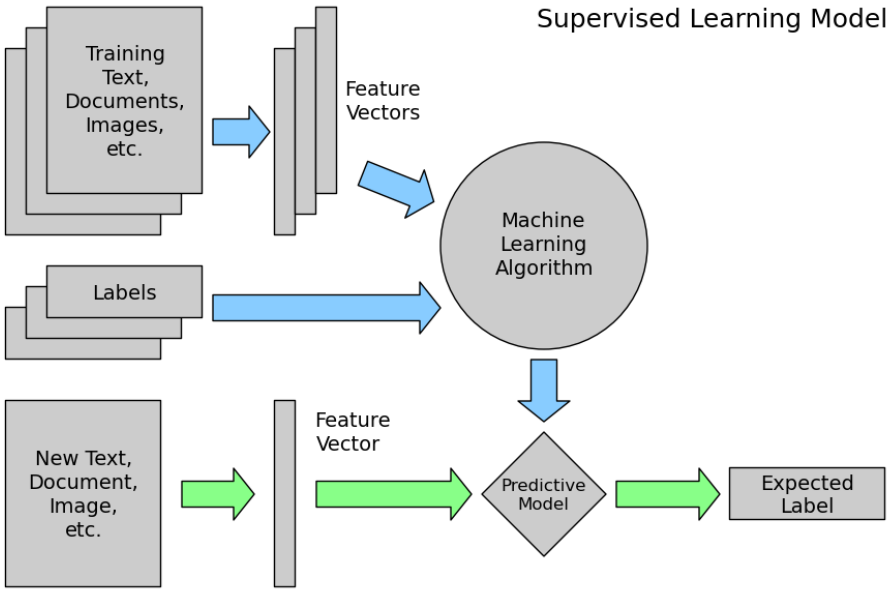

Figure 2.1: Supervised machine learning model [1]

2.1 Scikit-learn

Scikit-learn [3] is a Python library distributed as open source which provides a range of supervised and unsupervised machine learning tools for data mining and data analysis. It is built upon the SciPy library which must be installed before and includes:

- **NumPy:** Powerful N-dimensional array objects. It can be used as an efficient multi-dimensional container of generic data. This allows NumPy to seamlessly and speedily integrate with a wide variety of databases.
- **SciPy:** Fundamental library for scientific computing. It provides many user-friendly and efficient numerical routines such as routines for numerical integration and optimization.
- **Matplotlib:** Comprehensive 2D Plotting library which produces publication quality figures in a variety of formats and interactive environments across platforms.
- **IPython:** Enhanced Interactive Console with support for interactive data visualization and use of GUI toolkits.
- **Sympy:** Python library for symbolic mathematics
- **Pandas:** [4] Open source library that provides high-performance and easy-to-use data structures and data analysis tools for the Python programming language.

Scikit-learn provides tools for classification, regression and clustering problems but we are only interested in the classification ones as it is the problem that concerns to this project. It contains a great variety of classification algorithms with different characteristics which will allow us to make different tests to find the one with the best performance.

Here we can see the classification architecture in Scikit-learn:

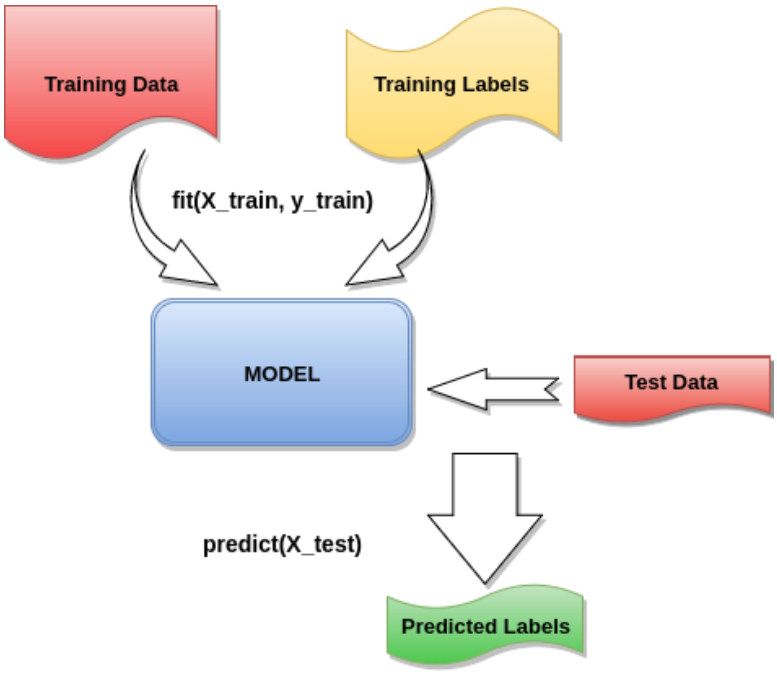

Figure 2.2: Model training and output prediction

2.2 Textblob

TextBlob [5] is a Python library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more. Particularly in this project, its use will be necessary for part-of-speech tagging.

Natural Language Processing is a field at the intersection of computer science, linguistics and artificial intelligence which aims to make the underlying structure of language available for analysis and manipulation with computer programs.

TextBLob is based on NLTK [6] and pattern libraries and it is compatible with Python 3, the version used in this project so it is perfect for it development.

2.3 Twitter API

The Twitter API provides programmatic access to read and write Twitter data. It is based on a REST architecture allowing the system to access and manipulate textual representations of Twitter web resources using a predefined set of stateless operations.

In this project it is used to obtain different kinds of information from the users that we are analysing. We can do that through a query with a GET petition including the parameters required in order to obtain the searched data. In response of this petition we get a JSON file consisting in a set of attribute-value pairs. This response has to be processed to extract the exact attributes we are looking for.

It is necessary to register as a developer in Twitter to get the credentials needed to have access to the API and it is very important to consider the limitations of this tool as it is only possible to make a limited number of petitions every 15 minutes. This number depends on the kind of request done and can be a crucial factor if the number of users or the variety of information required is big.

This tool has been used in this project to get the following information:

- The profile description of the user
- Any given URL in the profile description
- The number of followers
- The number of friends or people that the user is following

2.4 Senpy

Senpy [7] is an open source software developed by the GSI of the ETSIT-UPM. It is an implementation of a linked data model used for emotions and sentiments analysis. It is based on Marl, Onyx and NIF [8] semantic vocabularies and supports different formats as JSON-LD, Turtle, XML-RDF or simple text input. Despite our project is not related to sentiment analysis we are using this platform for the development of a plugin where we can deploy our system as a service with a web user interface. This way we can test and share our system easily.

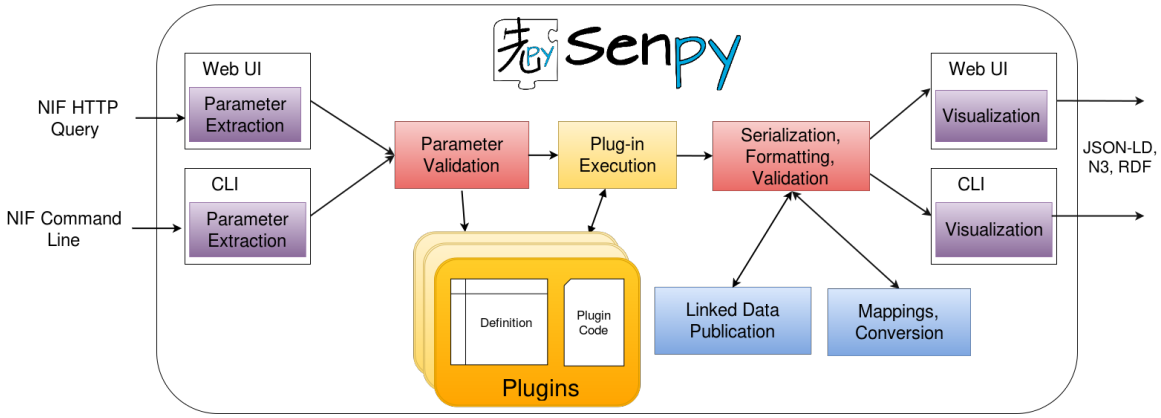

Figure 2.3: Senpy architecture model

In the *Figure 2.3* we can see the architecture of Senpy. It is formed by two modules, Senpy core, which is the building block of the service, and Senpy plugins, which consist of the analysis algorithm.

CHAPTER 3

Architecture

In this section, we are going to explain the general architecture of this project, including the design phase and implementation details.

First of all we are going to see a general view of the system in order to understand how it works without going into detail. To do it, the first step is the identification of the subsystems that are part of the project and the processes followed in its development.

Afterwards, every module will be explained in detail so we will have an overview of the system and a deeper vision that is going to make this project easy to understand.

3.1 Overview

In this section we are going to explain the general architecture of the project giving a brief description of the modules that are involved in the functioning of the system.

First of all, it is important to have a global vision so we are going to show how the different modules are interrelated in the *Figure 3.1*. From this schema we can proceed with the explanation of the function of the different modules and the operation of the whole system.

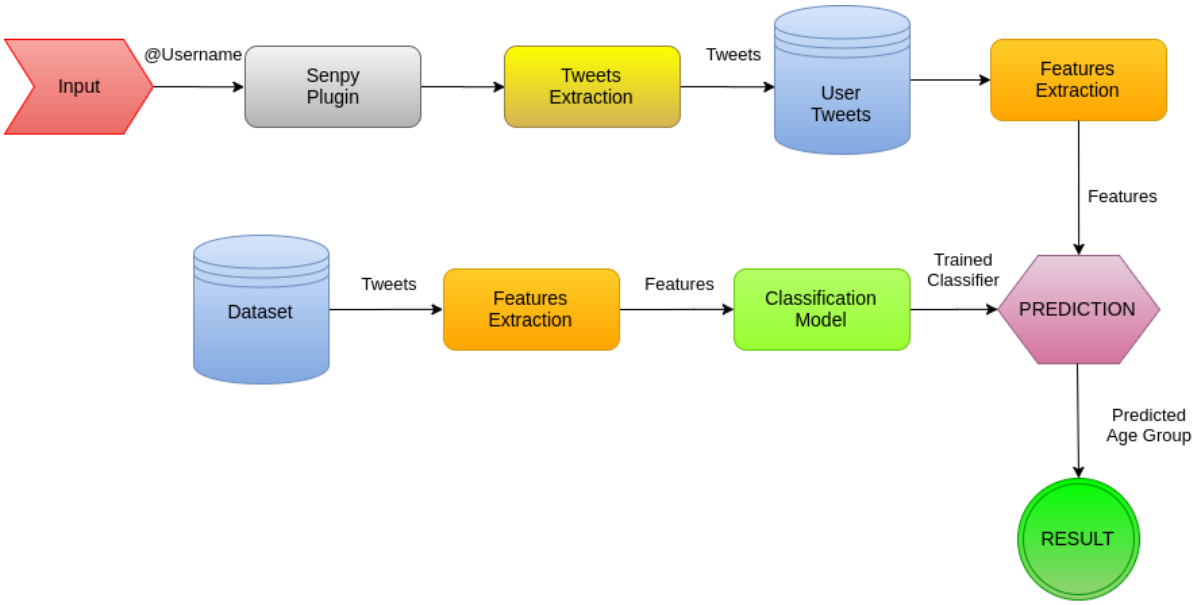

Figure 3.1: Architecture of the age classification system

As we have explained previously, the final goal of the project is to be able to predict the age of a Twitter user between five predefined age groups. This task is achieved through the development of a system based on machine learning techniques wrapped into a Senpy plugin so it can be used as a service.

The first step is to give as input of the system the Twitter username of the account we want to analyse. This username is used by the Twitter API to obtain a list of the last tweets shared by the user. This way we collect enough information in form of text to start processing it. In adittion to obtaining the last tweets of the user, the Twitter API is going to be used for the extraction of more relevant information as the profile description or the number of followers and friends that is going to be used in an extension of the system.

Once we have all the needed information from the user we need to process it to make it readable by the system. Here it is when the feature extraction module comes into operation. This module is the one in charge to transform the raw data obtained into useful information that can be analysed by a computer program. This is done transforming the data into vectors of quantifiable information, the "features". We have two groups of features used in different analysis that are going to be compared and later merged, the text features and the profile description features. The first of them are the ones extracted from the text of the tweets and the second ones use the text of the profile description in addition to other non textual features like the number of followers. This task is going to be explained in detail in *section 3.3*.

Until now it has been explained how to extract the needed information from the user to analyse, but all of this process is useless without the classification system. For the development of this part of the system we start from a labeled dataset with a set of classified users with their tweets. It is necessary to process this data to extract the username and the tweets separately. Then, we follow the same process as before extracting the features from this dataset that are going to be used as a learning source for the prediction algorithms. Once the features extraction is done, they are stored in a pickle file so we can save this step in the following executions, saving memory, time and computational resources.

Now we can feed the classification system with all this data. We are using different classification models and comparing them in order to achieve the best performance possible. The classification models used are explained in *section 3.4*.

At this time, we have to cross all the information obtained from the user with the classification model to elaborate a prediction.

All this work is wrapped into the plugin making the process very simple through the web user interface. It is only necessary to introduce the required username configuring the set of parameters as language and we will receive the prediction as response.

3.2 Extraction of the dataset information

The datasets used in this project has been obtained from the "PAN" platform [9]. In this platform, every year, several competitions related to machine learning topics are held. The topic in which we are interested is the author profiling competitions, from where we have obtained our datasets. These datasets are public and can be downloaded from this platform freely. We are using the datasets of the competitions held in the years 2016 and 2015 based on gender, age and other attributes identification. Each one of them provides data in Spanish, English and Dutch but we are going to explain its composition in detail in the corresponding section.

The general format of the datasets is almost identical with slight differences in the structure of the files. They, for every language, are composed by a set of XML files and one TXT file. Each one of the XML files belongs to a Twitter user and includes a list of tweets of this user and the username. The main difference between the datasets is that the one from 2015 does not include the username of the user. The text file, called truth.txt, is a list where every user of the XML files is matched with his age group so this is the labeled data used to train the algorithms.

These files are processed to obtain the username and tweets in a correct format so we can obtain the features used in the classification system independently of the dataset used. The extraction of these features is explained in detail in the next section.

Once we have extracted all the features that we want to analyse, we save the objects structure into a pickle file. These files provide a serializing and de-serializing system so we can save the data and use it later. This process is very helpful and saves a lot of processing time because we only extract the features once and they are available to be used at anytime.

3.3 Features Extraction

After the extraction of all the data from the files, with the tweets of the users in form of plain text, we have to transform this data into valuable information for a computer. It is very important to notice the differences of the kind of information that would be useful for a computer or a person. For a computer, unlike for a human, information like the username or the tweets in themselves are nothing more than text and can not make any inference from them. This is why it is necessary to process this text using language processing techniques in order to obtain valuable information for the computer.

The first step is to split the whole text into individual words and select the ones which are more useful in our analysis, the ones that provide some kind of information. This can be done using the TextBlob library which provides the tools needed for this task. The next step is to convert this selection of words into quantifiable information. This is done transforming it into vectors, containing all the information used by the classification system to perform the selection of the correct age group for every user. This vectors are what we call “features”.

In this section we are going to explain all this process in detail so we can understand how is it done. The extraction of these features is one of the most important parts of the classification system because they are the only information we have so without them our system would be useless.

Two kinds of features are going to be analysed in the following sections. The first of them are the features related to the text contained in the tweets. The second kind of features are the ones related to the user profile, like the description of the user or the number of followers.

3.3.1 Text features

In this section we are focusing on the features that can be extracted from the text of the tweets taking into account different aspects. For the election of the chosen characteristics to analyse we have been inspired by the work of other authors in this matter expanding their contributions.

We are going to list and explain the importance of the different text features chosen in this case for their use in age identification.

- **Capital letters:** The amount of capital letters used in the tweets as an indicator of age as there are differences in the use of them between the age groups. The number of capital letters is compared with the full length of the tweet.
- **Elongated words:** For example, the use of helloooo instead of hello. It is implemented as the number of words that have a sequence of the same character in themselves.
- **Exclamations and interrogations:** The amount of exclamation and interrogation marks denote differences in age. They reflect emotions and it is expected a higher use of them in the younger style. In this feature we have the count of this kind of items.
- **Punctuation marks:** The use of punctuation marks like commas or full stop. The grammar can be more neglected in some age groups so the number of punctuation marks is a feature to have in count.
- **Expressions:** There are some expressions very used in social networks, like “LOL” or “OMG” for example, which are associated with different age groups. The use of these expressions is quantified in this feature.
- **Laugh:** We measure the appearance of laugh expressions in English and Spanish, represented as the repetition of ’haha’ or ’jaja’ depending on the language. These expressions are often spelled incorrectly like ’aahhaha’. These cases are considered in this feature.
- **Emoticons:** The use of emoticons and smileys is a very important feature to have in count in age classification as it may be representative of younger age groups. Like in other features we count the appearance of these items.
- **Mentions, hashtags and retweets:** With this feature we measure the interaction with other users, wich may be different between age groups. We keep track of this kind of events by observing the appearance of ’#’, ’@’ or ’RT’ symbols.

- **URLs:** This feature counts the number of URL links in the tweets.As we see in the work of Schler [10], the number of hyperlinks increases with the age.
- **Pictures:** This feature counts the number of pictures. They appear in the tweets as hyperlinks so the explained in the previous feature is also applicable for them but we analyse them in a different category because they are a different interaction.
- **Average text and word lengths:** The study of the length of the whole text and the words of which it is composed can give us information for age classification. This feature measures the average length of this parameters.
- **Prepositions,determiners and pronouns:** As we can see in the work of Lucie Flekova and Iryna Gurevych [11], the use of this grammatical categories varies with the age. When we are getting older the number or preposition and determiners used increases and the number of pronouns decreases, so we make a comparison between the use of these groups.

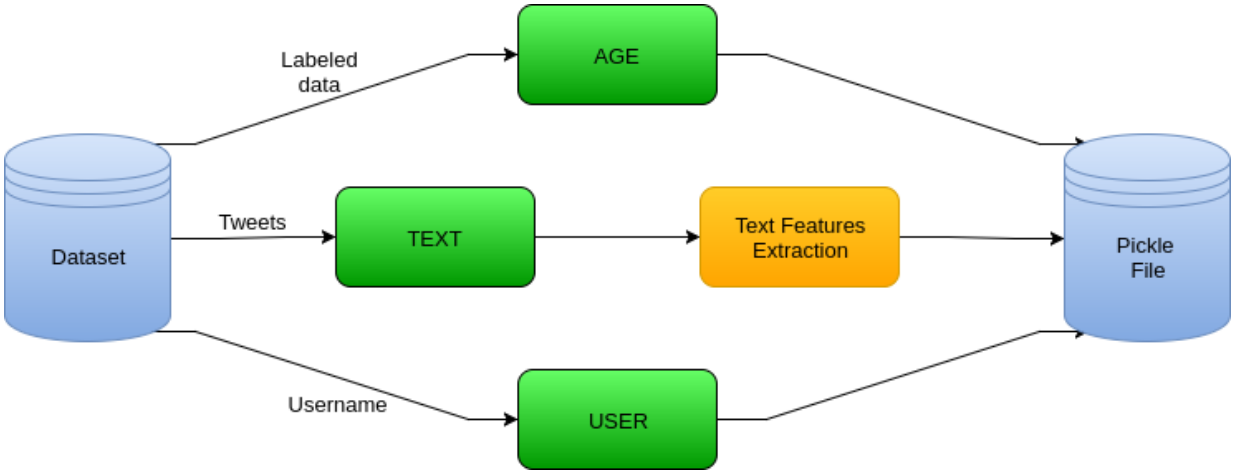

Figure 3.2: Text features extraction

3.3.2 Profile description features

This is the other group of features used in the development of the classifier. These features are based on the extraction of additional information from the user through the Twitter API using the username as reference.

- **Profile description:** In this case we obtain the profile description of the user through the API. The description text is processed and analysed following a process similar

to the one explained in the previous section. This way we extract the text features associated to the description of the user. The length of the description is much shorter than the list of tweets but it might present enough differences in a short format to be considered relevant.

- **Personal URL:** With this feature we check if the analysed profile has an associated personal URL like a blog or another social network.
- **Followers and following:** The ratio of followers/following reflects in some way how the user interacts in the social network. We extract the number of followers and following users with the API and calculate this ratio.
- **Number of tweets:** We measure the activity of the user in the social network with the total number of tweets registered.
- **Account creation date:** We can obtain the creation date of the account and extract the year of creation to compare the antiquity of the users.
- **Average tweets frequency:** With the number of tweets and the creation date we can easily calculate the number of tweets per day that the user posts so it can be used to measure the daily activity.

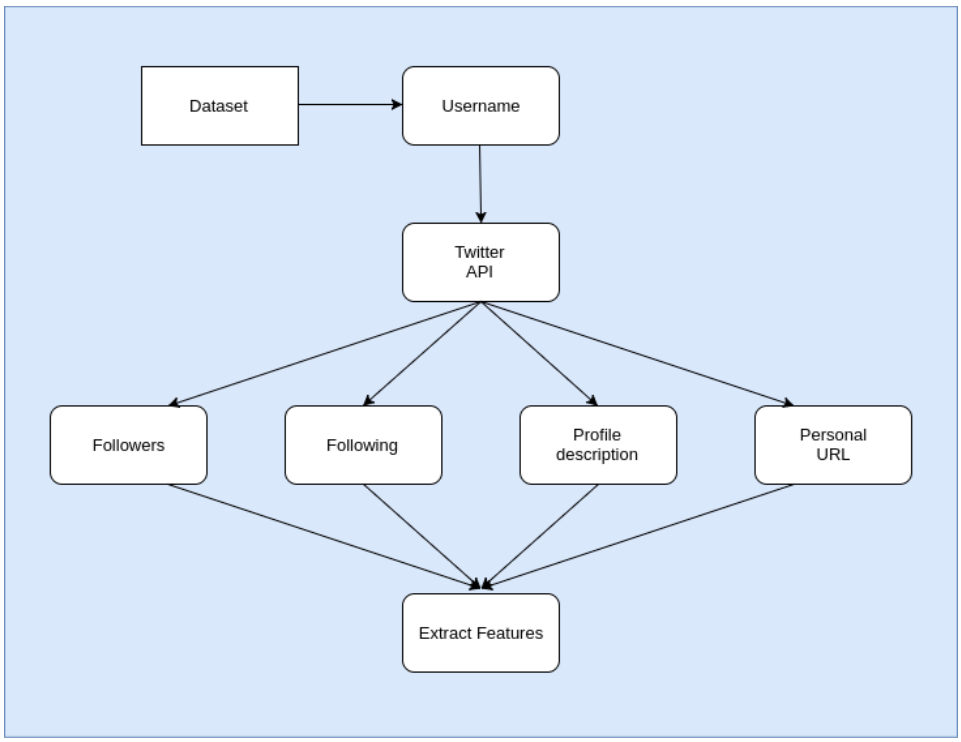

Figure 3.3: Profile description features extraction

3.4 Classification model

Once the features extraction task is finished we have to follow the next step, the development of the classification model. This task is based on the use of the extracted features to feed a classification algorithm that is going to learn from them and will be able to elaborate predictions for a new data entry based on the learned in the training process.

We have tested a set of different algorithms with their own particularities. This set of algorithms is described below:

- **K-Nearest Neighbors classifier (K-NN):** Neighbors-based classification is a type of instance-based learning or non-generalizing learning: it does not attempt to construct a general internal model, but simply stores instances of the training data. Classification is computed from a simple majority vote of the nearest neighbors of each point: a query point is assigned the data class which has the most representatives within the nearest neighbors of the point.
- **Linear model classifier:** It has been implemented using Logistic Regression which despite his name is a linear model for classification. In this model, the probabilities describing the possible outcomes of a single trial are modeled using a logistic function.
- **Random Forest classifier:** It is a meta estimator that fits a number of decision tree classifiers on various sub-samples of the dataset and use averaging to improve the predictive accuracy and control over-fitting. This particular one uses 100 estimators.
- **Decision Tree classifier:** Decision Trees are a non-parametric supervised learning method. The goal is to create a model that predicts the value of a target variable by learning simple decision rules inferred from the data features.
- **SVC:** This classifier uses Support Vector techniques. It uses a subset of training points in the decision function, so it is memory efficient.
- **Naive-Bayes Gaussian classifier:** The Naive-Bayes classifiers are a set of supervised machine learning algorithms based on applying the Bayes theorem with the assumption that every pair of features is independent. In this case it is assumed the features of the classes are distributed according to a Gaussian distribution.
- **Naive-Bayes Bernoulli classifier:** This is also a Naive-Bayes classifier but in this case it is assumed that the multiple features are binary-valued variables. This classifier explicitly penalizes the non-occurrence of a feature that is an indicator for a class.

- **Gradient Boosting classifier:** It builds an additive model in a forward stage-wise fashion. It enables the optimization of arbitrary differentiable loss functions. In each stage n-classes regression trees are fit on the negative gradient of the deviance loss function.

- **Ada Boost classifier:** This classifier consists in a recursive iteration of a learning classifier adjusting the weights of the samples in every iteration so the most difficult cases to predict have a bigger weight in order to focus on them.

- **One vs Rest classifier:** This strategy consists in fitting one classifier per class. For each classifier, the class is fitted against all the other classes. It computational efficiency is high and since each class is represented by one classifier, it is possible to gain knowledge about the class by inspecting its corresponding classifier.

Now we are going to describe the steps followed to put into practice the use of the classification system:

- **Division into training and testing sets:** For the evaluation of the algorithms it is necessary to split our data into two groups. One of them is going to be used for training the algorithm, as the learning source. The test group, as its name indicates, is used to test the performance of the system. For doing this splitting in an efficient way we have implemented a Stratified K-Folds cross validator using ten folds. This way we create stratified folds preserving the percentage of samples of each class.

- **Training:** In this step the system is fed with the training set. This is the way the prediction algorithms "learn" building relationships between the features and the different classes to study. Once the training process is finished the system is prepared to make the first predictions from new input data.

- **Testing:** The remaining part of the dataset that has been reserved for testing is used as input of the classifier. The system is going to classify this data into an age group based on what was learned in the previous step, analysing the features of this set.

- **Results:** The results obtained in the previous step are compared with the real results that we know from the *truth.txt* file. With this we can measure the performance of the algorithms from the percentage of success achieved.

3.5 Senpy plugin

This is the last part of the development of the system. All the work done so far, once a sufficiently good performance has been achieved, is merged into a unique functional service that is going to be used to elaborate new predictions. Thanks to Senpy we can deploy our system into a plugin and use it as a service with a simple and intuitive interface.

For the development of the plugin it is necessary to wrap all our code into one python file where all the system logic is found. Into this file there is one specific section where the training code is placed. This section allows us to store our classifier into a pickle file so we do not have to run the training process unnecessarily each time we make a prediction.

It is also necessary a plugin definition file with *.senpy* extension where all the attributes of the plugin are contained. Into this file we specify the libraries need by our system so they are loaded before the first run. The configuration parameters as the language that we are going to analyse are also specified into this file.

Figure 3.4: Age classifier Senpy plugin

CHAPTER 4

Evaluation of the Results

For the development of the system, a set of tests has been held obtaining different results that have to be interpreted correctly to evaluate the performance.

In this section we describe the characteristics of the tests performed and give a complete analysis of the results obtained.

The differences between the tests performed depend on the following characteristics:

- ***Language:*** *The performance of the system is different with every language chosen due to the grammatical differences between other characteristics and that some features have been programmed for a specific language.*
- ***Algorithm:*** *For the evaluation tests, we have chosen a set of classification algorithms which results may be very different. This is due to the characteristics of every algorithm and we must find the one that best suits to our system.*
- ***Features:*** *As described in Section 3.3, we have two sets of features so the performance will not be the shame depending on the one chosen.*

Finally, we will compare the results obtained with those obtained by other authors in similar works.

4.1 Datasets

In this section we are going to describe the characteristics of the datasets used in the project. For this task we are using two different datasets for every language evaluated so we can study not only the variations in the results obtained between different languages but also how the composition of the dataset affects them. All these datasets are composed by a list of xml files, each one corresponding to an author including a set of tweets, and a truth file in txt format which contains the labels of the age group of every author.

4.1.1 Dataset 1

This dataset was obtained from the "PAN" platform and dates from 02/29/2016. It follows the structure mentioned above with a set of XML files corresponding to each author. In *Figure 4.1* and *Figure 4.2* we can find the age distribution of the dataset for every language, observing a smaller number of samples of the older groups.

The characteristics of this dataset for every language are described bellow:

4.1.1.1 English

- Each file includes the **username** and the **tweets**.
- It contains a list of **436 users**.
- For each user the number of tweets collected is, as maximum, the last **1000 tweets**.

4.1.1.2 Spanish

- Each file includes the **username** and the **tweets**.
- It contains a list of **250 users**.
- For each user the number of tweets collected is, as maximum, the last **1000 tweets**.

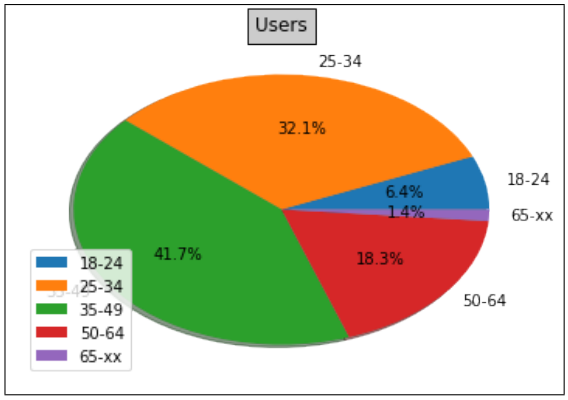

Figure 4.1: English distribution 2016

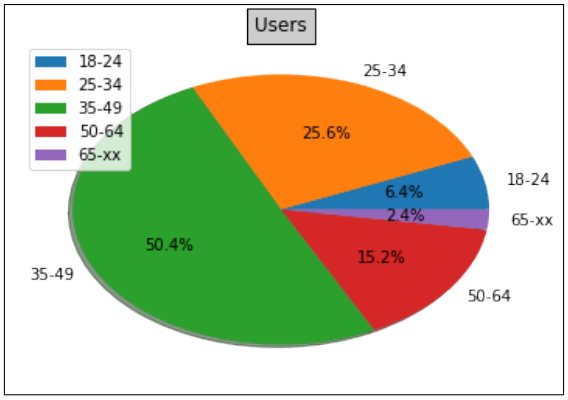

Figure 4.2: Spanish distribution 2016

4.1.2 Dataset 2

This dataset dates from 2015/04/23 and it is very similar but quite shorter than the previous one. The main difference we can find regarding to the dataset structure is that in this case we do not have the username so we are not going to use it for the profile description analysis because we do not have enough information to extract the features. Another important difference, as we see in *Figure 4.3* and *Figure 4.4*, is that in this case we do not have any sample for the older age group. This dataset provides less information but we are going to use it in order to have more variety of samples.

4.1.2.1 English

- Each file includes only the **tweets**.
- It contains a list of **152 users**.
- For each user the number of tweets collected is, as maximum, the last **100 tweets**.

4.1.2.2 Spanish

- Each file includes only the **tweets**.
- It contains a list of **100 users**.
- For each user the number of tweets collected is, as maximum, the last **100 tweets**.

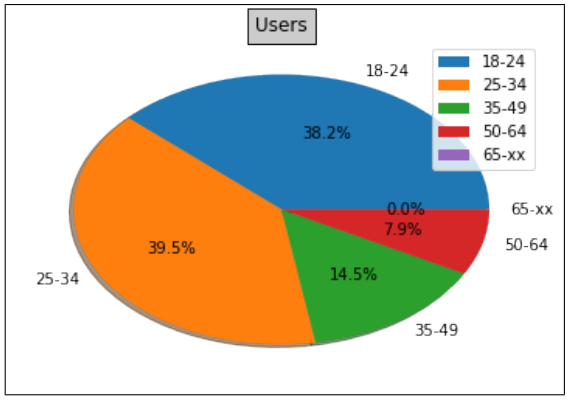

Figure 4.3: English distribution 2015

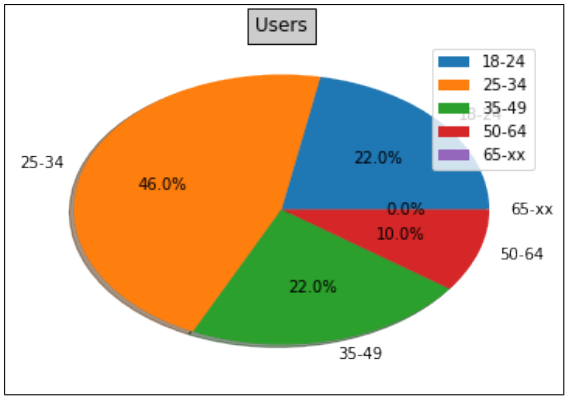

Figure 4.4: Spanish distribution 2015

4.2 Classification performance metrics

In order to evaluate the performance of the classification system there are different measurement functions that we are going to use. These metrics are very important and powerful because they are the indicators that allow us to know if the system performance is the expected. These metrics and their characteristics are listed below:

- **Confussion matrix:** This tool allows us to evaluate the performance in a quite graphical way through the representation of a results matrix. The diagonal elements represent the number of elements for which the predicted label is equal to the true label, while off-diagonal elements are those that are mislabeled by the classifier. The higher the diagonal values of the confusion matrix the better, indicating many correct predictions.

[H]

		Prediction outcome		
		p	**n**	**total**
actual value	**p′**	True Positive	False Negative	P′
	n′	False Positive	True Negative	N′
	total	P	N	

- **Accuracy:** This function compares the set of labels predicted with the true labels giving as result the percentage of success.

- **Precision:** It is the ability of the classifier not to label as positive a sample that is negative. Basicly it measures how right our classifier is when it says that an instance is positive counting the number of instances predicted as positives that were correctly evaluated.

- **Recall:** It measures the ability of the classifier to find all the positive samples. This counts the proportion of positive instances that were correctly evaluated.

- **F1-Score:** This is the main metric that we are going to use to evaluate the performance of the system. It is the harmonic mean of precision and recall, and tries to combine both in a single number and it reaches its best value at 1 and its worst score at 0. The relative contribution of precision and recall to the F1-Score are equal. The formula for its calculation is the following:

$$\text{F1} = 2 * (\text{precision} * \text{recall}) / (\text{precision} + \text{recall})$$

4.3 Model tuning

The proper configuration of the system parameters is a very important task if we want to obtain the best performance possible. In this case we are focusing in the choice of the appropriate number of features between all of them. The main reason for this feature selection is that if we feed our classifier with a big number of features it could cause the system to adapt to our dataset. If this happens, our classification system may have problems making predictions from new input data, being only able to work properly for specific inputs that have similar characteristics to the training set.

With the SelectKBest function, included in Scikit-Learn, we can choose the number of features that we want to use with the k parameter. This tool scores the features and removes all but the k highest scoring ones, so the features array will be reduced including only the most significant features. This will allow us to reduce the adaptation of the system to the training set and we will also get a slightly higher general performance.

For the election of the optimum value of the k parameter, we have tested the system with all the possible values comparing the accuracy obtained in every iteration so we can choose the value that gives us the best performance.

In *Figures 4.5, 4.6, 4.7 and 4.8* we can see an example of this process, observing the variation of the accuracy with the k parameter using the different datasets. As we see, the variation is very different in every case so we have chosen one k parameter for every one. For example, in *Figure 4.7*, the best performance is achieved at a value of k=12, improving the accuracy from 0.45 to 0.48, which supposes an improvement of more than a 6%.

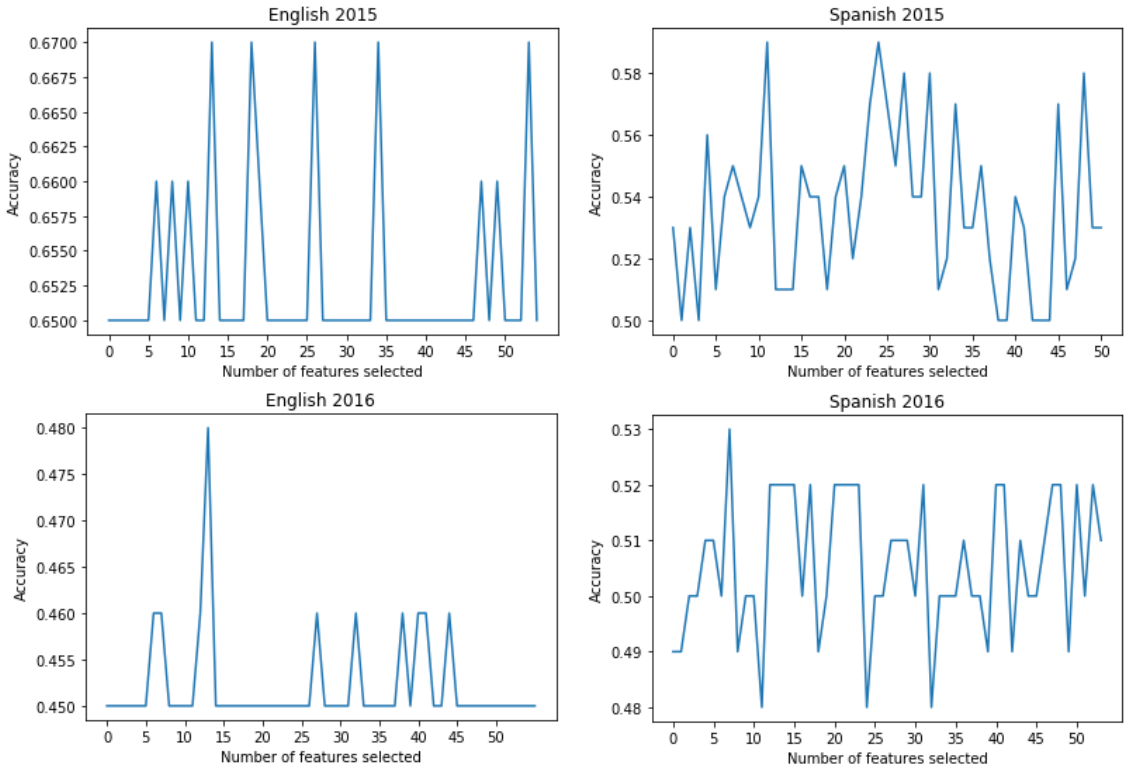

Figure 4.5: Feature selection

4.4 Tests

For the evaluation of the performance of the classification system, a set of tests has been developed. These tests has been designed to check how various factors can affect the performance and to find the optimal settings of the system for best results. The set of tests is listed below:

- **Test nº1:** Using both datasets in English and Spanish using only the text features to feed the classifier. With this test we can check the differences in the performance with the language and the number of users analysed. This is the only test that is

going to use the second dataset because it can not be used for the extraction of profile description features as it does not provide the username.

- **Test nº2:** In this one we are going to use only the profile description features with the dataset 1 in both languages. This way we can check the performance analysing only these features without the text of the tweets and compare it with the previous text.

- **Test nº3:** This final test merges all the development into one system using the text features and profile description features. This allows us to check if there is an appreciable improvement in the performance or, on the contrary, it is not worth it taking into account that the necessary computational resources will be considerably higher requiring a longer processing time.

4.5 Evaluation

The set of tests listed in the previous section is described in detail in this section, showing the results obtained in each of them and conducting a thorough analysis. After this, we will be able to compare the results with the work of other authors in this matter and extract conclusions about the performance.

The performance metric chosen for the evaluation of the following tests is the F1-Score and the results are going to be compared using different classification algorithms.

4.5.1 Text features evaluation

This is the first test that we are going to use to evaluate the performance of the system using only the text features. As described in the previous section, we are using both datasets 1 and 2, evaluating both of them in English and Spanish languages. This way we are going to obtain a set of varied results where we can analyse and interpret the differences regarding to the language and the composition of every dataset. These results will give us an overview of the behaviour of the basic system and in the following sections we are going to see how it changes with the attempts of improvement developed.

In first instance we are going to face this problem as a multi-label classification problem, where the system is going to try to identify to which age group each one of the analysed users of the testing set belongs. We are going to use the 10 different algorithms described in *Section 3.4* for each test so we can compare the results and choose the one with the best

performance.

In addition to these algorithms we are using two additional models, known as soft voting ensemble and hard voting ensemble. These ensemble models are based on the combination of the results of the different algorithms used but with slight differences between them. While the hard voting model chooses the class that has been predicted more frequently in the different algorithms, in soft voting this task is done taking into account the average probability for every class.

The results obtained are reflected in the *Table 4.1*.

As we see, the general performance is much better in the second dataset despite containing a much smaller number of users and tweets. We can also observe that the difference between the models used is significant and can reach a difference of 0.45 points between the best and the worst performance in the same test.

We were able to achieve a score of 0.82 with the second dataset in English, which is a very high result. The results in Spanish in this same dataset are slightly worse but can still be considered as quite good.

On the other hand, the results for the dataset 1 are not as good as the previous ones. In this case, the performance in Spanish is better than in English, but it can be improved since at no time it reaches the 50 % of success.

Table 4.1: F1-Score only with text features

Model	Spanish 1	English 1	Spanish 2	English 2
KNeighbors	0.37	0.24	0.41	0.70
Logistic Regression	0.44	0.31	**0.67**	**0.82**
Random Forest	**0.47**	0.32	**0.67**	0.66
Decision Tree	0.39	0.33	0.63	0.56
SVC	0.39	0.31	0.22	0.70
Gaussian NB	0.38	0.20	0.29	0.64
Bernoulli NB	0.44	0.26	0.33	0.62
Gradient Boosting	0.38	**0.38**	0.43	0.66
AdaBoost	0.35	0.36	0.47	0.71
One vs Rest	**0.47**	0.32	0.38	0.70
Soft Voting	**0.47**	0.37	0.63	0.70
Hard Voting	0.44	0.32	0.53	0.70

Now we are going to divide our system into 5 individual classification systems, one for each age group, facing the test as a mono-label classification problem. In this case every system is faced against only one class and has to determine whether or not the user belongs to that class.

In the *table 4.2* we can see the best result obtained for each dataset and language.

Table 4.2: Mono-label classification with text features

Dataset	**18-24**	**25-34**	**35-49**	**50-64**	**65-xx**
Spanish 1	0.94	0.73	0.75	0.86	1.0
English 1	0.93	0.63	0.61	0.80	1.0
Spanish 2	0.87	0.78	1.0	0.85	1.0
English 2	1.0	0.80	0.92	0.94	1.0

Although it may seem that the results in this case are excellent we must study them carefully since some may be misleading. This is due to the distribution of users for every age group. As we saw in *Section 4.1*, this distribution is not balanced and some groups, specially the older ones, have a very small number of samples compared to the total. That is why the system could predict that no user belongs to a determined class, so it would never be accurate when classifying a user belonging to this group but the F1-Score obtained is high because it is always correct when the user does not belong to this group. This can be seen clearly in the older group, as some of the datasets do not have samples for this group or the number of samples is minuscule.

4.5.2 Profile features evaluation

In this second test the only dataset that we are going to use is the dataset 1. As we explained before, this is because the second dataset does not provide the username and only identifies each list of tweets with a code so we can not use the Twitter API to extract the profile features.

We are only evaluating the profile description features to check how the system works with a small amount of information. In this case the test aims to obtain relevant information through new sources other than tweets. We have focused on aspects related to the user profile and other parameters, as the number of tweets, that allow us to measure the activity of the users.

As in the previous test we are comparing the performance of the different models and we can observe the results in the *Table 4.3*.

Comparing the results obtained for this dataset with the ones in the previous test we can observe that in this test there is a small improvement in both languages. We were able to reach a score of 0.41 in English, whereas in the previous test we reached only 0.38, which means an improvement of near 8%. In Spanish, we can observe that our results are better, reaching a score of 0.53 and getting over the barrier of 0.50 that could not be reached in the previous case. This score means an increase of 12.76% with respect of the previous text.

Table 4.3: F1-Score only with profile description features

Model	**Spanish 1**	**English 1**
KNeighbors	0.44	0.39
Logistic Regression	0.44	0.40
Random Forest	0.36	**0.41**
Decision Tree	0.39	0.30
SVC	0.44	0.26
Gaussian NB	0.17	0.28
Bernoulli NB	0.42	0.38
Gradient Boosting	0.47	0.35
AdaBoost	**0.53**	0.38
One vs Rest	0.44	0.16
Soft Voting	0.36	0.28
Hard Voting	0.44	0.36

Respect to the mono-label classification the results that we can observe in the *Table 4.4* are very similar to the ones obtained with the text features. Although it is true that the results of the groups with less users, in this case the older and the younger group, are not totally reliable, we can see that the results in the more significant groups are still quite good. For example, the most significant class is the "35-49" group, with the 41.7% and 50.4% of the samples in English and Spanish respectively. This class achieves scores of 0.55

and 0.64, being the performance in Spanish higher.

Table 4.4: Mono-label classification with profile description features

Age	**Spanish 1**	**English 1**
18-24	1.0	0.95
25-34	0.73	0.62
35-49	0.64	0.55
50-64	0.90	0.81
65-xx	1.0	1.0

4.5.3 Combined features evaluation

This test combines both sets of features as a sum of all previous work using the dataset 1 since by including the profile description features is the only one available. We are going to follow the same procedures as in previous tests. This way we can make a final comparison between the three evaluations and choose the one with the best performance.

In the Table 4.5 we can find the results of this test.

Once again, the best score has been obtained for the Spanish language with a value of 0.48 using the random forest algorithm. In English the maximum score achieved was 0.40 for logistic regression and gradient boosting.

We can observe a slight improvement respect to the results using exclusively the text features but the results are also worse than the ones with the profile description features. With these data on the table we can affirm that the combination of both models implies an improvement over the one only with text features, but not enough to overcome the use of only profile description features. For this reason we can conclude that the combination of features does not achieve the expected minimum results so it is not worth using and would be better to use the profile description features model since it achieves a higher score with the use of less computational resources.

Table 4.5: F1-Score with all features

Model	**Spanish 1**	**English 1**
KNeighbors	0.37	0.37
Logistic Regression	0.40	**0.40**
Random Forest	**0.48**	0.38
Decision Tree	0.28	0.28
SVC	0.26	0.26
Gaussian NB	0.26	0.26
Bernoulli NB	0.33	0.33
Gradient Boosting	0.38	**0.40**
AdaBoost	0.31	0.31
One vs Rest	0.41	0.34
Soft Voting	0.30	0.30
Hard Voting	0.30	0.32

Table 4.6: Mono-label classification with all features

Age	**Spanish 1**	**English 1**
18-24	0.95	1.0
25-34	0.62	0.73
35-49	0.55	0.67
50-64	0.81	0.90
65-xx	1.0	1.0

Since the results are very similar to the ones in the first test, in a mono-label classification we did not find any relevant variation as we see in the *Table 4.6*.

4.6 Related work

In this section we are going to present a comparison between our results and the ones obtained by other authors in similar experiments. This time the metric used is the accuracy since it is the one used by the other authors so we can compare the results.

We are going to compare with our accuracy results in both languages using the dataset 2 and only the text features.

Our results are summarized in the *Table 4.7* and the results of other authors in the *Table 4.8*

Table 4.7: Accuracy with text features using the dataset 2

Model	**English**	**Spanish**
KNeighbors	0.57	0.39
Logistic Regression	0.59	0.40
Random Forest	**0.65**	**0.54**
Decision Tree	0.48	0.45
Gaussian NB	0.59	0.42
Bernoulli NB	0.45	0.47
Gradient Boosting	**0.65**	0.52
AdaBoost	0.43	0.47

Table 4.8: Contributions of other authors. Datasets: (1): PAN 2013, (2): PAN 2014, (3): PAN 2015

Reference	Polarity	Model	Dataset	Acc. EN	Acc. SP
Flekova [11]	3 cat	Logistic Regression	(1)	0,53	0,57
Lim [12]	3 cat	SVM	(1)	0.48	**0.64**
Santosh [13]	3 cat	SVM / MaxEnt / Decision Tree	(1)	0.64	0.61
Marquardt [14]	5 cat	SVM	(2)	0.46	0.48
Villena [15]	5 cat	Multinomial Naive Bayes	(2)	0.37	0.48
Arroju [16]	5 cat	SGD	(3)	**0.69**	0.48

As we can see, the accuracy of our classification system is among the highest. In English our result is the second, with an accuracy of 0.68, only one hundredth below the best. Looking at the tables we might think that the result in Spanish is not that good but it is important to highlight that those values higher than ours have only used three age classification groups which makes the classification easier. Between the authors that used five classification groups we reached the best performance with a value of 0.54, six hundreds over the best accuracy achieved by other authors.

4.7 Senpy plugin evaluation

Once we have analysed the performance of the system objectively with the use of different metrics we can test how it behaves on the practice. For this, we are going to use the Senpy plugin, developed from our classification system.

We are going to analyse the Twitter profile of a user which age is known, and verify that the response generated by the plugin is the expected. The number of tweets that we analyse is limited by the Twitter API so we only obtain the last 200 tweets.

We have chosen as an example Elon Musk, who is 46 years old and we can see the response of the plugin in the *Figure 4.6*.

Figure 4.6: Senpy plugin response

As we see in the red box, our classification system gives the response as a FOAF [17] ontology which is a machine-readable ontology describing persons, their activities and their relations to other people and objects.

Through the classification system, the plugin has been able to correctly infer the age group to which the analysed user belongs, being in this case the '35-49' group.

This is only an example but after performing several tests in English and Spanish, our system has been able to correctly perform the classification in most cases.

CHAPTER 5

Conclusions and future work

In this last chapter we will describe the conclusions extracted from this thesis, problems faced and suggestions about future work.

5.1 Conclusions

In this section we are going to sum up all the conclusions into an overview of the project.

Our main objective was the development of an age classification system based on machine learning techniques, with a reasonable high accuracy, analysing the Twitter profile and tweets of different users. This objective has been fully achieved obtaining very good results in the classification and achieving a maximum F1-Score of 0.82 which is a really high result.

For the evaluation of the system we have extracted two groups of different features from the users according to different criteria.

The first group of features was related to the analysis of the tweets posted by the user, being the one that obtained the highest result,previously mentioned, of 0.82. This score was obtained using the algorithm of logistic regression for the English language and it is far above those obtained by other authors. In Spanish we also reached a good score of 0.67

using the text features so we can say the performance is very good for both languages.

With the second group of features we wanted to use a different source of information than the text of the tweets so, with the help of the Twitter API, we focused on the extraction of information related to the user profile in the social network as, for example, the number of tweets, the profile description or the number of followers. The results obtained with this group of features, despite analysing a much smaller amount of information, have been surprisingly better than expected even surpassing those obtained with the text features in some cases. This is an important fact since this module requires processing a much smaller amount of information so working with a large number of users its operation can be more efficient.

In last place, we have tested our development with the Senpy plugin, allowing us to make real-time predictions of users obtaining a direct feedback of the performance of the system.

With all this information we can infer that the performance of the system in general terms is very good considering the goals set at the beginning of the project.

5.2 Problems faced

During the development of this project we had to face some problems. These problems are listed below:

- **Twitter API limitations:** For the development of the profile description module we had to face some problems with the Twitter API. Tweeter limits the number of requests in a period of time which is a problem when working with a large number of users. There are several features that could not be included because of it since the number of requests required exceeded this limit.

- **Processing time:** Sometimes the features extraction process required a long processing time due to the large ammount of data analysed. This has not been the biggest problem since it only takes time to be solved but has produced a delay at the time of testing.

- **Senpy plugin:** It was necessary to adapt our code to the Senpy platform since it is designed for Sentiment and Emotion analysis and not for tasks as age classification.

5.3 Future work

In the following section I will explain the possible new features or improvements that could be done to the project.

New features: For the development of our system we have used two groups of features, the text features and the profile description features. With them we have obtained good results but there is still a great margin of improvement. One possible future line of work would be the research of new groups of features to analyse and the improvement of the ones that have been developed.

Use of a larger dataset: It would be interesting to use a dataset with a bigger number of samples since our datasets were not too big and had a very limited number of users in some of the age groups. With the use of a larger dataset we would have a greater variety and we could improve our results in those groups where we did not have a sufficient number of samples.

Improvement of results: In this thesis we have tested a set of different algorithms with different performance. The search for new models that are capable of obtaining better results would be a logical next step.

New profiling aspects: We have focused on the classification of users by age but we could take advantage of the knowledge acquired for the analysis of new profiling aspects as gender or personality.

Bibliography

[1] "Exercises for Intelligent Systems Course at Universidad Politécnica de Madrid, Telecommunication Engineering School." https://github.com/gsi-upm/sitc/blob/master/ml1/2_5_0_Machine_Learning.ipynb.

[2] T. Igarashi, J. Takai, and T. Yoshida, "Gender differences in social network development via mobile phone text messages: A longitudinal study," *Journal of Social and Personal Relationships*, vol. 22, no. 5, pp. 691–713, 2005.

[3] "scikit-learn: machine learning in Python." http://scikit-learn.org/stable/.

[4] "pandas: Python Data Analysis Library."

[5] "TextBlob: Simplified Text Processing." https://textblob.readthedocs.io/en/dev/.

[6] "Natural Language Toolkit — NLTK 3.2.4 documentation."

[7] "Senpy documentation." http://senpy.readthedocs.io/en/latest/senpy.html.

[8] C. A. Iglesias, J. F. Sánchez-Rada, G. Vulcu, and P. Buitelaar, "Linked Data Models for Sentiment and Emotion Analysis in Social Networks," in *Sentiment Analysis in Social Networks* (F. A. Pozzi, E. Fersini, E. Messina, and B. Liu, eds.), pp. 46–66, Morgan Kauffman, October 2016.

[9] "PAN official website." http://pan.webis.de/.

[10] J. Schler, M. Koppel, S. Argamon, and J. Pennebaker, "Enabling Management Oversight in Corporate Blog Space,"

[11] L. Flekova and I. Gurevych, "Can We Hide in the Web? Large Scale Simultaneous Age and Gender Author Profiling in Social Media Notebook for PAN at CLEF 2013,"

[12] V. L. L. T. Wee-Yong Lim, Jonathan Goh, "Content-centric age and gender profiling,"

[13] K. Santosh, R. Bansal, M. Shekhar, and V. Varma, "Author Profiling: Predicting Age and Gender from Blogs Notebook for PAN at CLEF 2013,"

[14] J. Marquardt, G. Farnadi, G. Vasudevan, M.-F. Moens, S. Davalos, A. Teredesai, and M. De Cock, "Age and Gender Identification in Social Media,"

[15] J. Villena-Román and J. C. González-Cristóbal, "DAEDALUS at PAN 2014: Guessing Tweet Author's Gender and Age,"

[16] M. Arroju, A. Hassan, and G. Farnadi, "Age, Gender and Personality Recognition using Tweets in a Multilingual Setting Notebook for PAN at CLEF 2015,"

[17] "FOAF Vocabulary Specification." http://xmlns.com/foaf/spec/.