Measuring Social Stress For Heart Attack Prediction

Gerard Casas Béjar

January 25, 2019

Polytechnic University of Catalonia

Signal Theory and Communications Department

Final Degree Project

Audiovisual Systems Engineering

Measuring Social Stress For Heart Attack Prediction

Gerard Casas Béjar

Co-advisors Marta Ruiz Costa-jussà
Carlos Escolano
Jordi Cortadella
Signal Theory and Communications Department
ETSETB
Polytechnic University of Catalonia

January 25, 2019

Gerard Casas Béjar

Measuring Social Stress For Heart Attack Prediction

Final Degree Project, January 25, 2019

Co-advisors: Marta Ruiz Costa-jussà

Carlos Escolano

Jordi Cortadella

Polytechnic University of Catalonia

Signal Theory and Communications Department

Barcelona

Abstract

There are many ways for predicting the stress in one single person but, nowadays, with the usage of social networks, there have appeared new ways for predicting the stress of the whole population. This project pretends to generate a system for predicting the heart attack probability based on the calculation of a measure of social stress. In this case, it has been used the Twitter 's trending topics as a social stress measure. The trending topics are the word or group of words more used by the Twitter's users in a finite period of time.

Abstract (different language)

Existen diferentes métodos para predecir el estrés en una persona pero, actualmente, con el uso de las redes sociales, han aparecido nuevas herramientas para predecir el estrés en la sociedad. Con este proyecto, se pretende generar un sistema de predicción del riesgo de infarto utilizando medidas de estrés de la sociedad. Concretamente, se han utilizado los Trending Topics de Twitter como la medida de estrés. Los trending topics son las palabras o grupos de palabras mas utilizadas por los usuarios de Twitter en un periodo de tiempo finito.

Acknowledgement

In this project, is recognised the enormous contribution of both the project co-directors Marta Ruíz and Jordi Cortadella in the search for solutions to the problems that have appeared during the project, as well as that of Carlos Escolano and his help in developing these solutions and proposing improvements to the project.

This project would not have been possible without the help of Dr. José Barrabés and Dr. Jordi Bañeras from the Vall d'Hebron Hospital and the data provided by the registry Codi Infart de Catalunya.

I would like to thank all the people who have been involved in this project to a greater or lesser extent: my family for the constant support, my teachers for the knowledge transmitted and my girlfriend for being by my side in the most important moments.

Dedicated specially to Martín Béjar Pérez.

Contents

Introduction 1

People's daily lifestyle is surrounded by external factors that produce emotional sensations, sometimes these factors produce well-being and other times they produce discomfort, anger and can affect their stress.

With the massive use of social networks, where people usually records their state of mind and their opinion about these external factors and other events that occur in their daily lives, a new door is opened for the study of the state of mind and feelings of society. Specifically, in this project, the use of trending topics allows us to know in a general way the interests or concerns of a specific geographic region.

Taking advantage of this tool that social networks include, we can study if there is any relationship between the different themes and other events that have occurred in the same period of time. In this specific case, the trending topics, that are recorded every day, have been used as a measure of stress and then related to heart attacks.

On the other hand, chronic stress is known to be a risk factor for heart attacks. But the main problem in associating attacks with stress is the cost of their measuring. With the use of the trending topics mentioned above, it would be possible to measure the stress of society according to the concept associated with them.

Using this data, it has been searched for the relationship between trending topics and heart attacks. In order to create a prediction system that allows us, based on the trending topics, to know the number of heart attacks. For this purpose, the typical machine learning algorithms have been used.

1.1 Statement of purpose

The main objective of the current project is implementing a prediction system based on machine learning algorithms using Twitter. To achieve this main objective it is required to: (1) perform the extraction of daily twitter trending topics using scraping tools and prepare the data for making the following computations; (2) group Twitter Trending Topics in order to reduce the dimension of the data, e.g. based on the area of the society they are related, i.e. Sports, Television or Politics; and (3), generate different graphics for obtaining a better visualization of the data and search for conclusions.

1.2 Requirements and specifications

Data with the heart attacks information is provided by register of ”Codi Infart de Catalunya”. This data has to be preprocessed in order to be used and is the only dataset with heart attack information that can be used. Also, it is required that for the development of scripts and models, the Python programming language must be used. This requirement gives the possibility to use all the already defined Python packages.

1.3 Methods and procedures

This project was firstly proposed by my co-directors in conjunction with two other projects that have been developed by Beatriz Pérez and Gerard Gallego. All of these projects are part of a collaboration with researchers of the Vall d'Hebron Hospital and share the data provided by the registry Codi Infart de Catalunya.

During the development of this project, there have been used different technologies and procedures which include web scrapping, Babelnet and machine learning. All of them defined in Chapter 2.

1.4 Work Plan

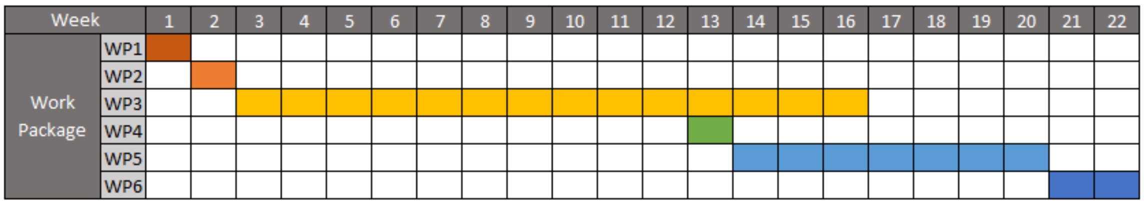
The whole project has been separated in different work packages and for each one there are included different tasks. Table 1.1 defines the work packages and their tasks.

WP	Task	Short Title	Major Constituent
1	1	Project Proposal and Work Plan	Documentation
2	1	Information Research	Documentation
3	1	Data mining and trending topic extraction	Development
3	2	Search for domains of trending topics	Development
3	3	Graphic and correlation measurements	Development
3	4	Implement a prediction system.	Development
3	5	Prediction improvements and tuning	Development
4	1	Critical Review	Documentation
5	1	Final Report	Documentation
6	1	Final Presentation	Documentation

Tab. 1.1: Work Packages and Tasks Summary

1.4.1 Gantt Diagram

Figure 1.1 shows the Gantt diagram. It defines the time distribution of tasks and work packages of the project. Concretely, it shows the tasks that have been developed throughout each week during the project.



Week		1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
Work Package	WP1	🟧																					
	WP2		🟧																				
	WP3			🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨	🟨						
	WP4													🟩									
	WP5														🟦	🟦	🟦	🟦	🟦	🟦	🟦		
	WP6																					🟦	🟦

Fig. 1.1: Gantt diagram that summarizes the time dedication to each work package.

1.5 Deviation and Incidences

During the development of the project there have been deviations from the initial idea and initial objectives. It should be noted that due to incidents there have been certain delays and modifications with respect to the initial planning.

In the first place, it should be noted that more time than expected has been devoted in the pre-processing of the data from Twitter. As will be commented in the section !! XX !!, the Trending Topic data is very inconsistent and difficult to deal with it. This factor has caused delays in the posteriors tasks.

On the other hand and consequently, it has been necessary to limit the analysis with the purpose of obtaining a prediction system as refined as possible.

Background

2

At this point, the background that holds the project is defined. Firstly, the heart diseases studies related with this one. After it, the knowledge representation is defined. Finally, the machine learning algorithms and how they work are described, going deeper into the specific algorithms used.

2.1 Heart Diseases

During the recent years there have been very significant studies in the fields of science related to the topics of this project. With the same objectives as this one, there are other projects that try to predict heart attacks with different datasets or search for correlations with other environmental data. In other cases, there are studies that look for health tasks by the use of social networks, for example: "*tracking illnesses over times (syndromic surveillance), measuring behavioral risk factors, localizing illnesses by geographic region, and analyzing symptoms and medication usage*" [PD].

On the other hand, looking for heart diseases and other pathologies there have been investigations that look, for example, for the relationship between alcohol consumption and arrhythmia and they found that: *Acute alcohol consumption is associated with cardiac arrhythmias and sinus tachycardia in particular* [Bru+].

On the other hand, searching for the relation between stress and heart attacks, studies have been carried out looking for the cardiovascular effects produced by environmental noise whose review provides evidence that "*noise not only causes annoyance, sleep disturbance, or reductions in quality of life, also contributes to a higher prevalence of the most important cardiovascular risk factor arterial hypertension and the incidence of cardiovascular diseases*" [Mün+].

Also, one of the recent investigations that has motivated this project has been the Jordi Bañeras investigation [Bañ+], which associates short-term exposure to air pollutants and the risk of ST elevation myocardial infarction.

All these projects show that there can be a relationship between stress and heart attack risk and also, there are different ways and measures related with the stress.

2.2 Knowledge Representation

"*In the information society, knowledge – i.e., the information and expertise needed to understand any subject of interest – is a key skill for understanding and decoding an ever-changing world. Much information is conveyed by means of linguistic communication (either oral or written), therefore it is critical to know how words are used to express meaning, i.e., we need lexical knowledge*" [NP]. In order to develop a classifier based on text, it is necessary to have a system that allows the appropriate interpretation of the text. In this section, BabelNet is described as the essential tool used for text interpretation.

BabelNet is a very large, widecoverage multilingual semantic network developed by Roberto Navigli. "*BabelNet's key approach is the integration of lexicographic and encyclopedic knowledge from WordNet and Wikipedia. In addition, Machine Translation is applied on it to enrich the resource with lexical information for all languages*" [NP].

BabelNet encodes knowledge as a labeled directed graph. It calls the multilingually lexicalized concepts as *Babel synset*. It harvests the concepts and relations from the largest available semantic lexicon of English, WordNet, and a wide-coverage collaboratively-edited encyclopedia, Wikipedia:

- From WordNet, all available word senses (as concepts) and all the lexical and semantic pointers between synsets (as relations).
- From Wikipedia, all encyclopedic entries (i.e., Wikipages, as concepts) and semantically unspecified relations from hyperlinked text.

This semantic network, also gives the possibility to get the *domain* of each word. A domain is a semantic range, a way of classifying each concept in a group depending on its sense and the context. For example, if it is searched for "Lionel Messi" in BabelNet, the resulting domain would be "*Sports and Recreation*".

2.3 Machine Learning

Machine learning is defined as a field of data science and a branch of artificial intelligence, whose objective is to identify automatic decision techniques, applicable to various problems arising in different areas or working disciplines. More specifically, the goal is to create programs that can detect behaviors from information provided as a training. The most common tasks done with machine learning algorithms are classification and prediction [DHS02] [Bis06].

A machine learning development consists of three parts:

1. Preprocessing: The dataset is prepared for being classified by the use of filters.

2. Selection of characteristics: It is possible that not all the data provided by the dataset has to be used in the algorithm or many of them are lineal combinations that do not give more information about the register.

3. Automatic decision algorithm: Selection of the algorithm or combination of algorithms that will be used as a classifier.

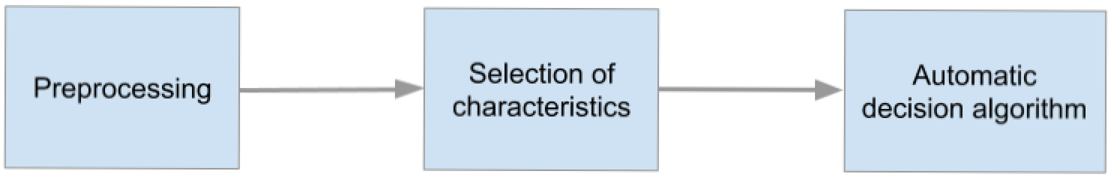


Fig. 2.1: Machine learning process.

Three types of machine learning algorithms are distinguished depending on how its training is defined

1. **Supervised learning:** the machine is provided with desired outputs and the aim is to produce the correct output when given a new input.

2. **Unsupervised learning:** the aim is to build a model that can be used to classify or define a phenomenon without having a labelled database.

3. **Learning by reinforcement:** the machine produces actions that affect the state of the environment, and receives rewards (or punishments). Its objective is learning how to act in a way that maximizes long-term rewards.

In this project there have been used different supervised machine learning algorithms because there was an available labeled dataset. In the following section, the algorithms used are described:

Random Forest

Random Forest is a particular way of bagging[1] where different decision trees are trained with different subsets of features. When those trees are in the learning step, they found

[1]Algorithm to reduce the instability on hard-to-classify patterns that combines weak classifiers and generate a decision by voting.

the best way of separating the dataset depending on their feature values and they define the leaf nodes where the feature values decide the final class. Due to the random feature selection, the decision trees are more independent from the others. One of the advantage is that is faster in the training process, because each tree learns only a subset of these features. Some parameters that can be optimized are:

- Number of trees of the forest.
- Minimum number of samples for splitting an internal node
- Minimum number of samples for being a leaf node.

Multilayer Perceptron

The Multilayer perceptron or MLP is an artificial neural network that contains different layers with a group of neurons. There are three kind of layers depending on its position:

- **Input layer:** This layer is the responsible of transferring the input values to the rest of the network. The neurons of this layer do not process the data.
- **Hidden layers:** This layer or layers contain the neurons that process the patterns that are introduced as inputs.
- **Output Layer:** This layer generate the output value of the system.

.In the picture 2.2 is shown the schema of a Multilayer Perceptron and the defined layers. MLP uses the back-propagation technique that implies that, during the training, the error is calculated from the output layer back to the input layer, and that this is what allows the model to be trained with more than one hidden layer

In this algorithm there are some parameters that can be optimized depending on its function:

- Optimization method: stochastic gradient descent, Adam, LBFGS...
- Learning rate.
- Hidden-layers size.

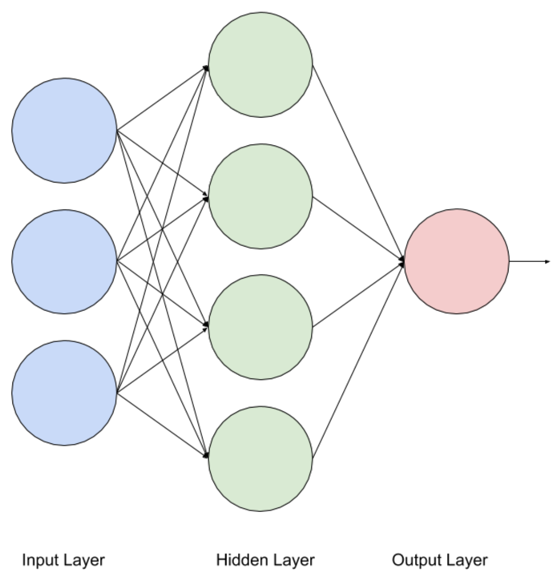


Fig. 2.2: Multilayer perceptron structure

Support Vector Machine

Support Vector Machines belong to the category of linear classifiers, since they induce linear or hyper-flat separators, either in the original space of the input, if the classes are separable, or in a transformed space, if they are not separable. But that only happens if it is not used a kernel, or the kernel is linear. A kernel is a transformation that is applied to the data to represent them in a space in which they are separable. SVM classifiers look for the separator that has the maximum distance with the points that are closer to it.

The support vectors are those that are the most nearly to the decision boundary. In the picture 2.3 there are seen the support vectors defined for an SVM classifier with two classes. To implement a SVM that allows classifying in N classes, one-by-one SVM is performed: one classifier is trained for each possible pair of classes and the one with the highest probability of being of that class is selected.

In the SVM there are different parameters that can be optimized:

- C: The penalty parameter of the error term.
- Kernel type: linear, polynomic, RBF...
- Maximum number of iterations.

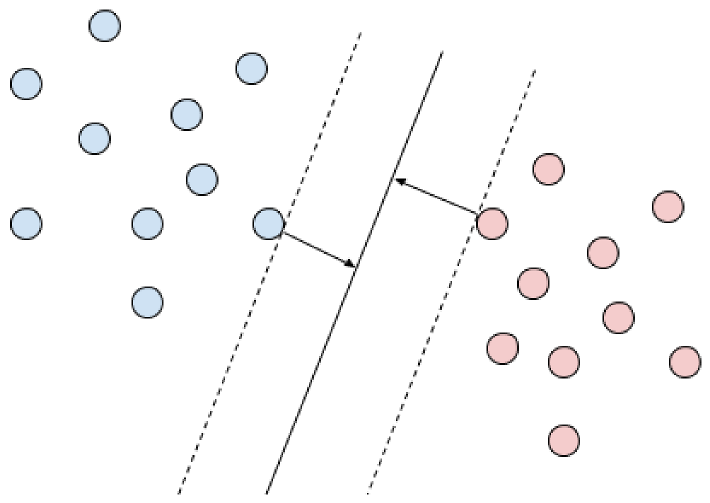


Fig. 2.3: Support vectors defined for an SVM classifier with two classes

K-Nearest Neighbours

The K-Nearest Neighbours (KNN) is a supervised classification method where the distance between the input and the training dataset's vectors gives the result. For a given k, the resultant class would be the most common from the k nearest vectors. This kind of learning it is also called "lazy learning" because the function is only locally approximated and the whole computation is deferred to the classification. In the picture 2.4 is defined an example where the green vector would be classified as the blue class. In this type of classifiers there is only one parameter that can be optimized that is the number k of nearest neighbours to count.

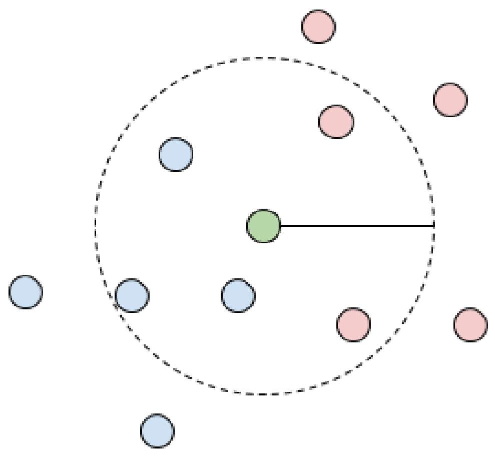


Fig. 2.4: Example of KNN for a given $k = 5$.

AdaBoost

AdaBoost or Adaptive Boost is an meta-estimator that combines different weak algorithms outputs in a weighted votation.

It begins by fitting a classifier on the original dataset and then fits additional copies of the classifier on the same dataset but with the incorrectly classified instances so that other classifiers are focused on the more complex instances. In the picture 2.5 is defined as an schema the algorithm aplied for AdaBoost.

As the other algorithms, there are parameters that can be optimized in the AdaBoost algorithm:

- Number of estimators
- Learning rate
- Type of estimator

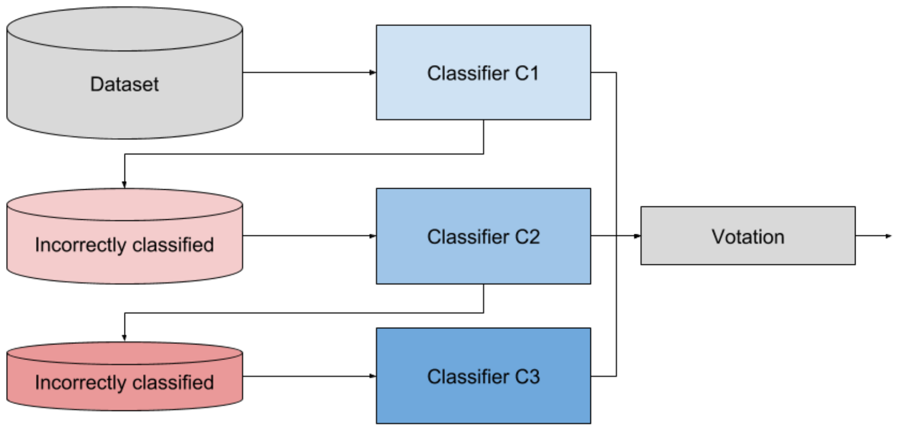


Fig. 2.5: AdaBoost algorithm.

Evaluation

There are different measures to evaluate the quality of a classifier. Most of them use this values:

- *True Positive (TP)*: For a given class, is the number of inputs correctly classified on it.
- *False Positive (FP)*: Is the number of inputs incorrectly classified in a given class.

- *True Negative (TN)*: Is the number of inputs correctly classified out of the given class.
- *False Negative (FN)*: Is the number of inputs incorrectly classified out of the given class.

With these values, different quality measures can be obtained within a classifier:

$$Precision = \frac{TP}{TP + FP} \tag{2.1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2.2}$$

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.3}$$

$$Fscore = 2 \cdot \frac{Precision \cdot Recall}{Precision + recall} \tag{2.4}$$

In this project these measures are used for comparing the results of the different algorithms applied with a given labeled dataset.

3 Methodology and Implementation

Taking into account the objective of defining a classifier capable to predict information about heart attacks produced in Barcelona based on the information of the Twitter trending topics, there has been defined the following methodology and explained the implementation of it.

3.1 Methodology

Before explaining the procedure carried out, it is worth highlighting the different parts involved in the project. Since the point in which the data is obtained until the acquisition of the measures of quality that allow to evaluate each classifier. As is defined in 2.3 there are three parts in the development of a prediction algorithm. In this project is defined a new part: the evaluation which make available to select the best algorithm with the previous functions defined in 2.3. In preprocessing are defined 3 steps: Data acquisition, tune-up and transformations. After this, there is required an step of features selection for the classifiers. After this, there is a step for the automatic decision algorithms, where is included the hyperparameters selection for each exercise done using cross-validation and the algorithms comparison. At the end, the final evaluation is defined for obtaining the presented results for the classifiers quality measures. The figure 3.1 shows the schema of the methodology applied.

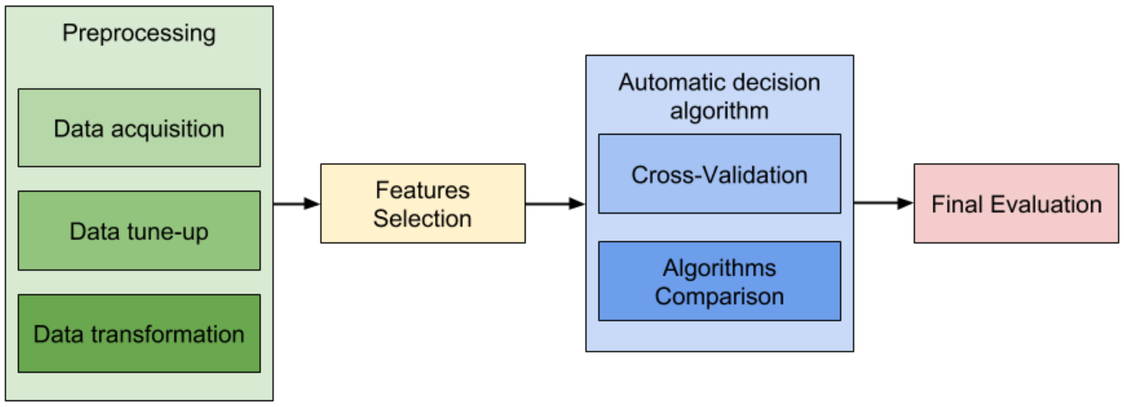


Fig. 3.1: Methodology applied in the project implementation

3.2 Preprocessing

In this part of the project are implemented all the previous steps required for developing a classifier. For creating a classifier, the first step is to define a ready-to-train dataset where all the information that would be used for classifying has to be defined.

Data acquisition

The part of the data corresponding to heart attacks has been provided by ”Codi Infart de Catalunya”. In the dataset, are defined cases of heart attack that have been recorded in the province of Barcelona from the year 2010 to 2016. Specifically, this study is carried out with daily data of acute myocardial infarctions with ST elevation [1]. This dataset contains the characteristics defined in 3.1, in it, each row of the table represents one case of heart attack.

Characteristic	Value
Recorded cases	17.000
First Date	01-01-2010
Last Date	31-12-2016
Maximum daily cases	18
Geographical area	Province of Barcelona

Tab. 3.1: Characteristics of the heart attack dataset provided by Codi Infart de Catalunya

But there isn’t any free dataset with the recorded trending topics in Barcelona between these years. So, the first part of the project is the trending topic data acquisition. There are different ways to obtain data. One of them is known as parsing or web scrapping, where data is extracted directly from the source code of the web page where it is located. There is a website that contains this kind of information so a parsing algorithm has been implemented for this purpose.

Being python one of the requirements for the development of the project’s scripts, it is worth mentioning the python libraries that allow to do this. One of the most famous libraries is Beautiful Soup 4. This library allows to extract data from HTML or XML files. Beautiful Soup 4 is a Python library for pulling data out of HTML and XML files. This library provides simple methods for generating a parse tree in the minimum lines of Python code. Nowadays, as the other famous Python Library for doing this process, HTML Parser, Beautiful Soup 4 is the most common and used library for Web Parsing. In the algorithm 1 is defined the followed procedure for parsing data from `trendogate.com` where, for a given geographical area and date, returns the trending topic list. In this website, the date and the geographical

[1]The ST segment represents the beginning of ventricular repolarization and corresponds to the slow plateau repolarization phase of ventricular myocytes.

area are defined in the URL so, changing the components of date from the URL requested, the trending topic of each day can be obtained. One of the consequences of using this procedure is that in Trendogate are just recorded information from the 21-02-2015 to the present date so, it implies that the database of heart attacks and trending topics only share two years so all useful data will be found between these two years.

```
Input  : Trending topics URL (trendogate.com)
Output : JSON with the dictionary of trending topics for each date.
Initialize dates dictionary
for each date do
    Format the url
    Obtain the html for the given url
    Initialize a soup as 'html.parser'
    Initialize the daily topics array
    for each 'list-group' do
        for each 'list-group-item' do
            Insert value in daily topics array
        end
    end
    if length(daily topics array) != 0 then
        Insert daily array into date dictionary
    end
end
Generate JSON with the dates dictionary
```

Algorithm 1: Algorithm for obtaining the trending topics from trendogate.com for each date

With this algorithm around 14,000 trending topics have been obtained, but many of them are not ready to be interpreted by any system because of its format. So the next step will try to transform this trending topics into words.

Data tune-up

In this part the dataset is prepared for its purpose. Firstly there has to be performed a grouping of the heart attacks and the information of the trending topics of the same day. Using *csvkit*, it is easy to group the information for date and have the count of registers for each date value.

On the other hand, Twitter's trending topics have a particular format that needs to be changed in order to use text processing techniques. The most used format in trending topics is defined by some easy rules:

1. Starts with a '#'

2. There are not white spaces between different words.

3. Each different word, starts with uppercase.

One example of Twitter trending topic would be : '#HappyMonday'. These rules are not always complied in all trending topics and this implies that some of them would not be easily interpreted by text processing tools or in the worst case should be omitted. For doing this task it has been implemented the algorithm 2, where it deletes the #ánd inserts an space between a lower case followed by an upper case.

```
Input   : JSON with the dictionary of trending topics for each date.
Output : JSON with the "clean" version of the input.
input = load JSON
for key in input do
    for trending in range(0,length(input[key])) do
        input[key][trending] = input[key][trending].replace '#' for nothing
        leave = False
        while not leave do
            nextpos += 1
            if nextpos == length(input[key][trending]) then
                leave = True
            end
            if input[key][trending][nextpos-1] is not UpperCase and input[key][trending][nextpos] is UpperCase then
                insert space in input[key][trending][nextpos]
            end
        end
        lstrip on input[key][trending]
        replace input[key][trending] in input[key]
    end
end
Generate a new JSON with clean data
```
;

Algorithm 2: Algorithm for cleaning the trending topic dataset.

Data transformation

With a group of 17,000 trending topics where most of them are different but preserve the same meaning, it has been necessary to make a transformation into groups according to the meaning of the trending topics. In the first place, it was proposed to obtain the hyperonym of the trending topic words. Algorithm 3 shows how to do this step using BabelNet 2.2 and sending requests to its pages. It is necessary to take into account that BabelNet has a limit of calls to its pages, initially 1.000. But, in this project by sending a request, it was possible to increase the call limit to 45.000. In this algorithm is necesary to call BabelNet two times. The first one for obtaining the sense of the trending topic and the second one for obtaining the hyperonyms.

```
Input   : JSON with the "clean" version of the input.
Output : JSON with the dates, the trending topics and their hyperonyms.
input = load JSON
Define output dictionary; keycounter = 45.000
for key in input do
    Define dictionary of hyperonyms
    for trending in range(0,length(input[key]) do
        keycounter = keycounter - 2
        if keycounter == 0 then
            Generate a new JSON with data
            exit
        end
        Set BabelNet key
        service_url definition
        lemma = input[key][trending]
        request and response
        if length(response) == 0 then
            hyperonyms[lemma] = "null"
        end
        else
            Get words with relation of hyperonyms
            service_url definition
            Insert words in hyperonymsfound
            hyperonyms[lemma] = hyperonymsfound
        end
    end
    output[key] = hyperonyms
end
Generate a new JSON with dates, the trending topics and their hyper-
  onyms
```

Algorithm 3: Algorithm for obtaining hyperonyms from the trending topics.

After the implementation of this algorithm it has been seen that these hyperonyms seem to be as dispersed as the original trending topics, so it was necessary to take another option.

For a correct and definitive grouping it has been used the *domain* field obtained in the *synset* of the a BabelNet response for each trending topic. So, in this case the previously defined algorithm 3 has been changed for the following algorithm 4:

```
Input  : JSON with the "clean" version of the input.
Output : JSON with the dates, the trending topics and their domains.
input = load JSON
Define output dictionary; keycounter = 45.000
for key in input do
    Define dictionary of domains
    for trending in range(0,length(input[key]) do
        keycounter = keycounter - 2
        if keycounter == 0 then
            Generate a new JSON with data
            exit
        end
        Set BabelNet key
        sense_service_url definition
        lemma = input[key][trending]
        Request and obtention of the response
        if length(response) == 0 then
            domains[lemma] = "null"
        end
        else
            synset_service_url definition
            Request and obtention of the response
            domains[lemma]= list(response[domains].keys())
        end
    end
    output[key] = domains
end
Generate a new JSON with dates, the trending topics and their domains
```

Algorithm 4: Algorithm for obtaining the dictionary of domains from the trending topics.

With this algorithm there has been generated an amount of 6.000 trending topics with their domains correctly defined. This means that the other 11.000 trending topics could not be related to a domain. This can be caused by two things:

1. The given trending topic has not been found within BabelNet.

2. No domain has been found for a given trending topic.

With the objective of prediction and training a classifier it is very important to have as much data as possible. So in this part the human interpretation takes part in order to insert in the correct domain the not-defined trending topics. For doing this implementation it has been implemented a system for obtaining the most common words seen on not-defined trending topics of the output JSON of the algorithm 4 in order to replace their domains manually. In the algorithm 5 is defined the way for obtaining the most common words of the not-defined trending topics.

```
Input   : JSON with the dates, the trending topics and their domains.
input = load JSON
Define null_words array
for key in input do
    for key2, value in input[key] do
        if value == "null" or value == "[]" then
            null_words += key2.split()
        end
    end
end
Define Counter with null_words
Sort items of Counter
for i in range(len(Sorted_items)) do
    print(Sorted_item[i])
end
```

Algorithm 5: Algorithm for obtaining the most common words of the not-defined trending topics.

In this step the most common words in the non-defined trending topics are known. The next objective would be to replace the null domains and the non defined trending topics with an existing domain by searching for those words and replacing the domain field. As can be seen in the table 3.2 there are 3 main domains that are introduced for the null trending topics with the algorithm 6.

With this algorithm, the amount of null domains is reduced from 11.000 to 7.000 topics. This corresponds to an amount of 10.000 trending topics with an assigned domain. This data will be enough for training and designing a classifier.

3.3 Features Selection

With a clean data set, it is time to make a selection of the appropriate parameters to train the prediction system. The first idea proposed is to generate a daily vector with bag of words(BoW) format for trending topics [2]. This would generate a vector of 17.000 positions for each day, where for each day only 20 to 50 trending topics will be reported and the rest of positions will be assigned the value 0. Using this kind of vectors has the advantage that the not-defined trending topics are used to but also this suppose that the noisy trending topics such as "#HappyMonday" or "#IsMyBirthday" that have no relationship with stress are

[2]Where each column represents a trending topic and the value is informed to 1 if that trending topic appears this day

Words	Domains
GH, belen, EH, TV3, L6N, L6, chiringuito, Voz, Gala, First Dates, MTV, zapeando, TVE, cambiame, cámbiame, Mediaset, Divinity, Video, MVT, Motiv, AVis, Premios, Premio, Pelis, Debate, ARV, Media, Actualidad, Salvame, Final, Walking, Film, A3, myhyv, Express, Eurovision, Chachi, Canal, Show, Gemeliers	MEDIA
Ciudadanos, Podemos, PSOE, Partido Popular, Puigdemont, Política, Politica, España, Spain, Democracia, Democratica, Parlament, colau, Rajoy, Pedro, Rivera, Vota, República, Pleno, Iglesias, Villalobos, Rato, Aguirre, Anti, Irak, guerra, Barcenas, War, Brexit, Justicia, Libertad, Catalu, Contra, Voto	POLITICS_AND_GOVERNMENT
Messi, Riera, Cristiano, UCL, Futbol, Copa, Liga, Gol, Betis, Bulls, NBA, Alves, Vietto, Bale, Pique, Villarreal, Lull, Mundial, Ganar, Sporting, Stegen, Keylor, Mathieu, Ciclism, morata, casilla, Depor, Dybala	SPORT_AND_RECREATION

Tab. 3.2: Most common words and their assigned domain.

```
Input   : JSON with the the dates, the trending topics and their domains.
Output  : input JSON including manually assigned domains
input = load JSON
Define media_words and media_domain
Define politics_words and politics_domain
Define sports_words and sports_domain
for key in input do
    for key2, value in input[key] do
        if value == "null" or value == "[]" then
            if key2 in media_words then
                input[key][key2] = media_domain
            end
            if key2 in politics_words then
                input[key][key2] = politics_domain
            end
            if key2 in sports_words then
                input[key][key2] = sports_domain
            end
        end
    end
end
Generate a new JSON with the modified dictionary
```

Algorithm 6: Algorithm for inserting domains to the not-defined trending topics.

included in the training dataset. In this project, the choice made has been the following: It has been generated a kind of daily bag of words with the domain information but including the percentage of its domain in the daily trends instead of a binary value. The number of different domains is 35, defined in the table 3.3 what is so much lower than the number of trending topics. In addition, the noisy trendings that would be included in the BoW of trending topics, would be collected inside the "null" category. For this task it has been implemented the algorithm 7 where also includes the daily heart attack information in the output CSV. In this algorithm it was used the package *pandas* for working with the DataFrame objects.

```
Input  : JSON with the dates, the trending topics and their domains.
Input  : CSV with the dates and the heart attacks.
Output : CSV file with the BoW with percentage of each day.
heartregister = load CSV file
domainsregisters = load JSON file
Define a output Dataframe df with the Date, NumHeartAttacks and BoW
  columns
i = 0
for key in domainsregisters do
    day_bow = [0]*35
    df.loc[i] = [0]*37
    Transform csv datetimes in same format as JSON input file.
    df.xs(i)['Date'] = key
    register = heartregister[(date == key)]
    numatcks = register['Num Infartos'].get_values()
    df.xs(i)['Infartos'] = int(x)
    Count number of domains in numdomains for each the date including
      nulls and '[]'
    for key2 and value in domainsregisters[key].items() do
        if value == 'null' or value == '[]' then
            df.xs(i)['null'] += 1/numdomains
        end
        else
            for each val in value do
                df.xs(i)[v] += 1/numdomains
            end
            i += 1
        end
    end
end
Save output dataframe as CSV
```

Algorithm 7: Algorithm for obtaining the CSV with Daily heart attacks and percentage BoW of domains.

With this CSV there have been obtained many graphics for interpreting the results. Firstly there is defined the Probability density function of the daily heart attacks registered for the days between 21-02-2015 and 31-12-2016 in the 3.2. As it can be seen in the figure, the mean number of daily heart diseases is between 6 and 7. This is the baseline for detecting anomalies in other graphics.

Position	Domain Name
0	SPORT_AND_RECREATION
1	MUSIC
2	null
3	TEXTILE_AND_CLOTHING
4	MEDIA
5	GEOGRAPHY_AND_PLACES
6	TRANSPORT_AND_TRAVEL
7	POLITICS_AND_GOVERNMENT
8	GEOLOGY_AND_GEOPHYSICS
9	ROYALTY_AND_NOBILITY
10	GAMES_AND_VIDEO_GAMES
11	BUSINESS_ECONOMICS_AND_FINANCE
12	LITERATURE_AND_THEATRE
13	PHILOSOPHY_AND_PSYCHOLOGY
14	HEALTH_AND_MEDICINE
15	LANGUAGE_AND_LINGUISTICS
16	RELIGION_MYSTICISM_AND_MYTHOLOGY
17	WARFARE_AND_DEFENSE
18	BIOLOGY
19	FOOD_AND_DRINK
20	METEOROLOGY
21	PHYSICS_AND_ASTRONOMY
22	COMPUTING
23	EDUCATION
24	HERALDRY_HONORS_AND_VEXILLOLOGY
25	ART_ARCHITECTURE_AND_ARCHAEOLOGY
26	LAW_AND_CRIME
27	ANIMALS
28	FARMING
29	CHEMISTRY_AND_MINERALOGY
30	MATHEMATICS
31	HISTORY
32	NUMISMATICS_AND_CURRENCIES
33	ENGINEERING_AND_TECHNOLOGY
34	CULTURE_AND_SOCIETY

Tab. 3.3: Relation between position and domain value.

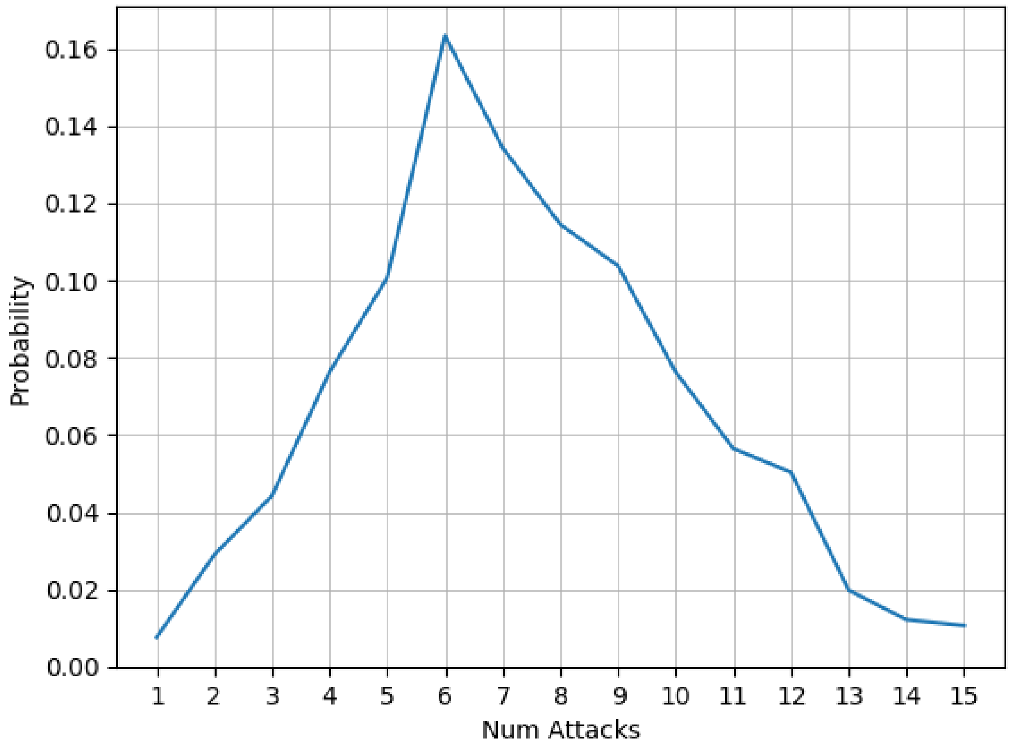


Fig. 3.2: Mean number of registered heart diseases for each domain.

In the figure 3.3 can be seen the mean heart attack registers for each appearing domain, not including the last domain "CULTURE_AND_SOCIETY" because it has been defined as an outlier index because of its low representation and just contains disorientating data. In addition, the 8th domain "GEOLOGY_AND_GEOPHYSICS" where most of the trending topics included in this domain are related to environmental catastrophes, shows an increase of more than 1 point above the average.

As the most common domains: Media, Sports and Politics, there have been included more graphics looking for any visible relationship between heart attacks and these specific domains. In addition, the same graphics have been made with the domain GEOLOGY_AND_GEOPHYSICS because of the high mean value seen on the 3.3. On the following graphs it can be seen the mean percentage of appearances of each domain for each number of infarcts. This can give a better resolution of how each domain interact with the number of heart attacks. This can be seen in the figure 3.4. It can be seen that the pattern varies for values above twelve recorded cases. For example in the case of the politics domain 3.4b, the tendency is increasing until the 13 registered instances, above this value, acts in a different mode. It happens in a similar way with the sports domain 3.4c, where the tendency is decreasing until the 12 registries. This can be produced because of a low amount of data for knowing this kind of relations. On the other hand it can be seen that the domain of

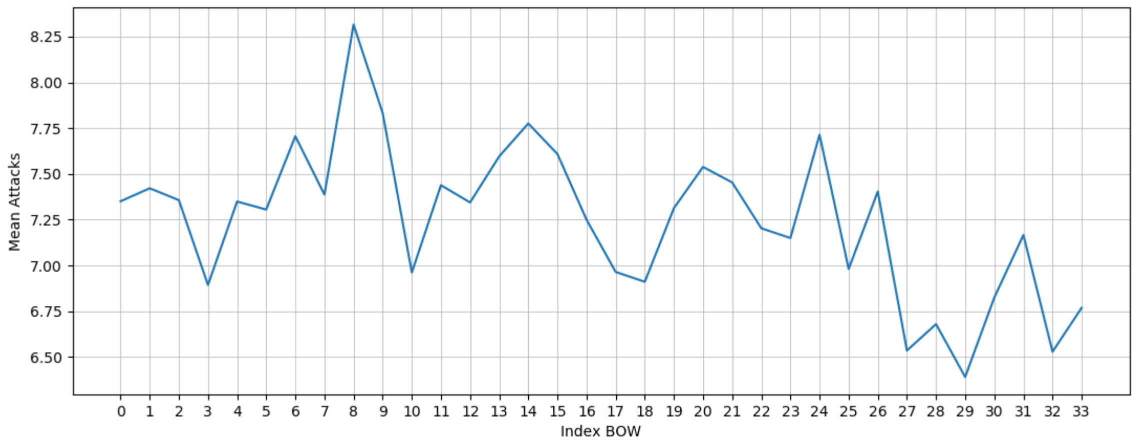


Fig. 3.3: Mean number of registered heart diseases for each domain.

geophysics acts in a very peculiar way because it can be approached by a step in 15 heart attacks. This may be due to the fact that it does not appear as much as other domains. Probably, with a greater amount of data, a better graph could be obtained.

After observing the relationship between the percentages of domains and the number of heart attacks, it has been tried to validate that these values are representing properly the reality using the standard deviation of each percentage. As it can be seen in the image 3.5, the deviation is very high in the whole range, which means that the values are very overlapped and this can cause worse results in the prediction.

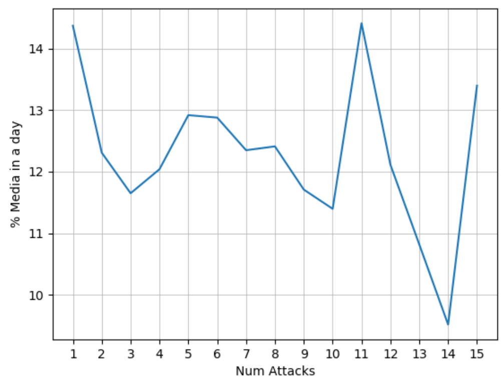
3.4 Automatic decision algorithm

In this part, the different algorithms defined in the chapter: 2.3 are trained in order to predict information about daily heart attacks. There have been defined two different exercises of prediction:

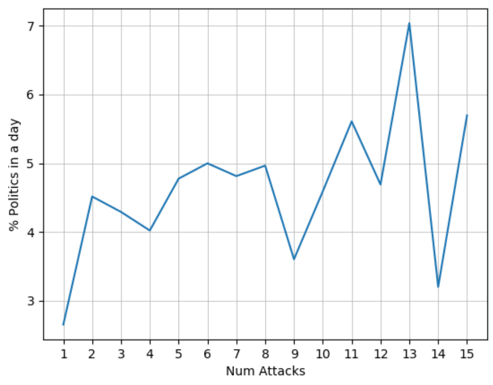
1. Predict for a given domain list, if the number of heart attacks will be in the mean, above or below it.

2. Predict with the domains and the number of heart attacks of the previous day, if there will be more or less heart attacks compared to its predecessor.

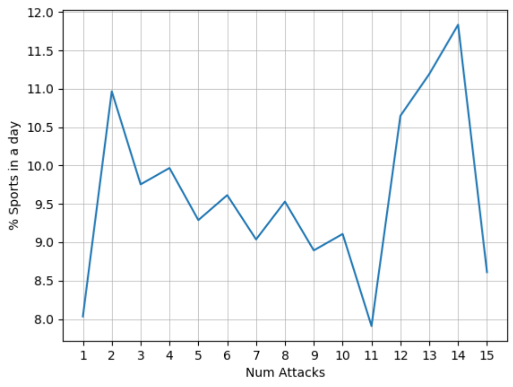
For this purpose, the dataset has been separated in three parts. Training, Dev (or Validation) and Test. The training dataset, 80% of the dataset, has been used for training the predictor and obtain the best hyperparameters using the cross-validation method. The Dev, as a 20% of the training dataset, has been used for comparing the different prediction algorithms



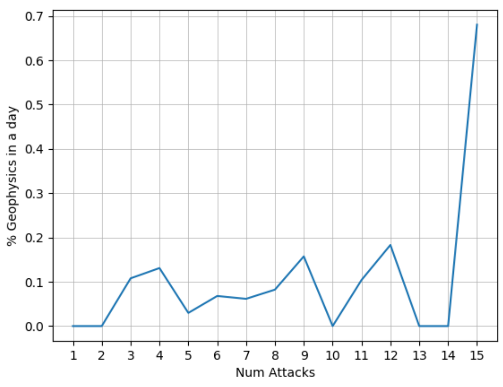
(a) Media



(b) Politics



(c) Sports

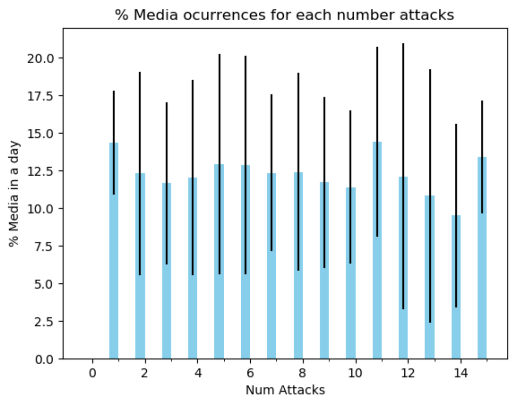


(d) Geophysics

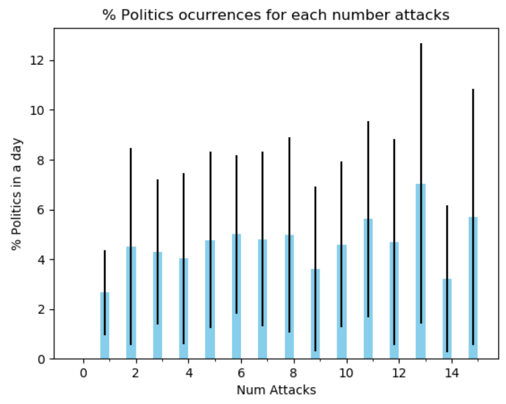
Fig. 3.4: Mean percentage of each domain appearing for each number of heart attacks

as defined in the figure 3.1 and finally, the test dataset, a 20% of the dataset, gives the final accuracy and F-score value for the chosen predictor. It should be noted that for machine learning developments, the *sklearn* library has been used [Ped+]. For obtaining the hyperparameters it has been used *GridSearchCV* using cross-validation method. All the hyperparameters have been obtained with the same algorithm defined in 8.

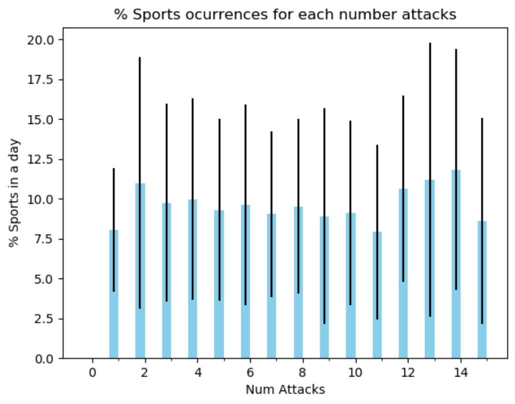
```
Input  : CSV format dataset.
Output : Hyperparameters of the classifier.
Separate features and labels
Full dataset Train/Test split (80% and 20%)
Train dataset Train/Dev split (80% and 20%)
Select the model
numparameter = np.arange(minvalue, maxvalue, step)
gscv = GridSearchCV(model,
  param_grid= dict(parameter = numparameter), n_jobs = -1)
gscv.fit(train_features,train_labels)
print(Best Value)
```
Algorithm 8: Algorithm for obtaining the hyperparameters for a given model.



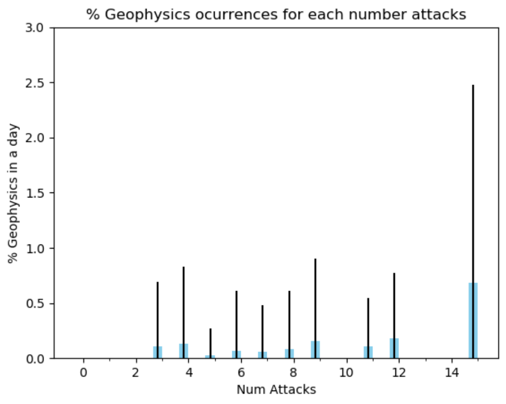
(a) Media with STD



(b) Politics with STD



(c) Sports with STD



(d) Geophysics with STD

Fig. 3.5: Mean percentage of each domain and standard deviation appearing for each number of heart attacks

Prediction of categories

For this development, an algorithm has been made that defines the three classes dividing the value of the number of heart attacks, which serve as labels, by 5 and try to predict this value. In order to get better results it has been used PCA[3] algorithm for reduce the dimensionality of the dataset.

After this, the hyperparameters have been defined for each used model using algorithm 8 and at the end they have been compared using the Dev dataset. In the algorithm: 9 this procedure is defined. With this algorithm has been obtained a set of results defined in 3.4:

[3]PCA is a technique used to describe a data set in terms of new non-correlated variables. Components are ordered by the amount of original variance that is described on them. This technique is useful for reducing the dimensionality of a data set.

```
Input   : CSV format dataset.
Output : Quality measures for each classifier.
Separate features and labels
labels = labels/5
Full dataset Train/Test split (80% and 20%)
Train dataset Train/Dev split (80% and 20%)
PCA transformation (n_components = 0.99) in features
for each model do
    Select the model
    Cross-Validation for obtaining the best hyperparameters
    Model definition with best hyperparameters
    prediction = model.predict(dev_features)
    print(classification_report(test_labels, prediction))
    print(accuracy_score(test_labels, prediction))
end
```
Algorithm 9: Algorithm for obtaining the quality measures for each model.

Model	Hyperparameters	Accuracy	F-Score
Random Forest	*n_estimators = 800* *min_samples_split = 15* *min_samples_leaf = 10*	55,3%	0.27
Multi-Layer Perceptron	*max_iter = 10000* *solver = 'adam'* *hidden_layer_sizes = 100*	57,6%	0.24
AdaBoost	*n_estimators = 150* *learning_rate = 0.4* *base_estimator = 'Decision Tree'*	58,4%	0.27
K-Nearest Neighbours	*k = 19*	56.9%	0.28
Support Vector Machine	*max_iter = 1000* *C = 0.1*	57,6%	0.24

Tab. 3.4: Results obtained for each model and their hyperparameters chosen in the categories exercise.

As you can see, the results are not really good. The value of F-Score is not in any case above 0.3 and the accuracy is always below 60%. It is worth mentioning that the algorithm that obtained the best F-score was K-Nearest Neighbours with 0.28 and the worst were MLP and SVM. On the other hand, for the accuracy value, the maximum was obtained with AdaBoost with 58.4% and the minimum with Random Forest with 55.3%.

In addition, it has been observed that SVM and MLP classify all results in the same class. This is because of, on the one hand, the way the data are distributed 3.2 and, therefore, that the database is not correctly balanced and, on the other hand, the insufficient amount of data. Due to these poor results, it has been decided to discard this exercise and not to apply the test data to these predictors.

Prediction compared with the previous day

For this exercise we have had to add two more columns to the database: the first, with the information on the number of heart attacks of the previous day and the second with the label that informs if the number of heart attacks has grown or decreased. To do this, the information from the top row of the database has been taken and added to the given row. One of the most important differences between this and the previous exercise is that the number of possible labels has decreased from 3 to 2. The algorithm used in this section is identical to 9 but selecting the value of the grown or decreased column as the label. Also, in this case the PCA transformation has been removed because it gives worse results. In the table 3.5 the results are shown.

Model	Hyperparameters	Accuracy	F-Score
Random Forest	*n_estimators = 500* *min_samples_split = 16* *min_samples_leaf = 20*	76%	0.75
Multi-Layer Perceptron	*max_iter = 1000* *solver = 'lbfgs'* *hidden_layer_sizes = 100*	74,2%	0.74
AdaBoost	*n_estimators = 120* *learning_rate = 0.01* *base_estimator = 'DecisionTree'*	76%	0.75
K-Nearest Neighbours	*k = 3*	65,7%	0.65
Support Vector Machine	*max_iter = 10000* *C = 0.9*	65,7%	0.64

Tab. 3.5: Results obtained for each model and their hyperparameters chosen in the "grow or decrease" exercise

As it can be seen, the results in this case are so much better and are ready for testing. As it can be seen, for the chosen database, contrary to the previous exercise, the KNN F-score is among the lowest together with SVM, this one with the worst F-score: 0.64. On the other hand, the best results have been obtained with Random Forest and AdaBoost. In them, it has been obtained a 0.75 F-score and a 76% accuracy. These results seem to be good. These results are much better than those obtained in the previous exercise. In this case, the final test has been performed with the test dataset made in the algorithm 9.

3.5 Evaluation

For the evaluation process, a comparison has been made with the predictor performed in another project. Specifically, it has been compared with the *grow or decrease* predictor developed by Beatriz Pérez, which uses data from the bourse. To make this comparison, a dataset common to both predictors was used. For this, a series of rules had to be added due to the combination between the two datasets.

1. The test has to defined between 21-02-2015 and 31-12-2016 because are the first and last date in the trending topics dataset.

2. The test can not include dates of the weekend because there is not information about bourse in the dataset on Saturday or Sunday.

After transforming the dataset, there has been done a prediction. The obtained results are defined in the next table 3.6

Model	**Dataset**	**Accuracy**	**F-Score**
RandomForest	**Bourse**	76.9%	0.76
	Trending Topics	75.2%	0.74
AdaBoost	**Bourse**	76.2%	0.75
	Trending Topics	75.2%	0.74

Tab. 3.6: Comparison between the prediction based on data from the bourse and the trending topics from Twitter.

As it can be seen, the results obtained with the predictor developed by Beatriz Pérez, are a little better than those obtained with the trending topics. This may be due to the amount of data that has been used to train one and other predictor. In the case of the bourse the dataset data has not been limited to be contained between 2015 and 2016 instead all available data from the database of heart attacks has been used, even though the data of the weekends had to be removed.

It is interesting that the two predictors trained with trending topics return the same results for both Random Forest and AdaBoost, but it is even more curious that the same happens with the Dev data observed in the table 3.5. In any case, it can be seen that with this type of data it is possible to predict simple information about the heart attacks that are going to occur with acceptable confidence.

4 Conclusion

In this chapter are defined the extracted conclusions of the whole project. There is defined an specific part for the possible improvements of the system and the personal opinion of the author.

To begin with, it is concluded that a simple information predictor about heart attacks can be made from Twitter's trending topics. Although it is not entirely clear if a relationship can be established between stress and trending topics, the relationship between stress and heart attacks can be mentioned. This is not to suggest that exist a causality between these two factors. It is important to note that the trending topic is a type of data very difficult to handle because it is formed by many words (sometimes impossible to separate) but do not provide context by themselves.

4.1 Possible Improvements

Different improvements can be made to the design besides collecting more data about trending topics. For example the validation of more hyperparameters or the use of other types of predictors can be carried out. The option of training complex neural networks has been discarded due to the low amount of data, but it would not be strange to perform the same exercise with a complex neural network or to apply deep learning.

Other improvements can be to preprocess the trending topics by using more rules and try to clean them better. In the same way that it is possible to combine the answers obtained by different semantic networks like BabelNet, and to use other interpretation systems for the trending topics that contain more words which BabelNet does not categorize so well.

Finally, it is proposed to improve the use of all the twits associated with these trending topics in order to offer the context they do not have by themselves.

4.2 Personal opinion

On a personal level, it has been very interesting to develop this project because the objective is very beautiful and useful for people. On the other hand, it should be noted that the most difficult part has been the preprocessing of the trending topics where in a moment there was no solution to exploit the data. Thanks to the co-directors help, it has been possible to solve those problems as much as possible and obtain quite remarkable results.

Bibliography

[Bañ+] Jordi Bañeras, Ignacio Ferreira-González, Josep Ramon Marsal, et al. *Short-term exposure to air pollutants increases the risk of ST elevation myocardial infarction and of infarct-related ventricular arrhythmias and mortality*. URL: https://www.internationaljournalofcardiology.com/article/S0167-5273(17)34210-9/fulltext. (accessed: 13.01.2019) (cit. on p. 4).

[Bis06] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006 (cit. on p. 5).

[Bru+] Stefan Brunner, Rebecca Herbel, Cathrine Drobesch, et al. *Alcohol consumption, sinus tachycardia, and cardiac arrhythmias at the Munich Octoberfest: results from the Munich Beer Related Electrocardiogram Workup Study (MunichBREW)*. URL: https://academic.oup.com/eurheartj/article/38/27/2100/3748448. (accessed: 13.01.2019) (cit. on p. 4).

[DHS02] R. O. Duda, P. E. Hart, and D. G. Stork. *Pattern Classification*. Wiley Interscience, 2002 (cit. on p. 5).

[Mün+] Thomas Münzel, Tommaso Gori, Wolfgang Babisch, and Mathias Basner. *Cardiovascular effects of environmental noise exposure*. URL: https://academic.oup.com/eurheartj/article/35/13/829/634015. (accessed: 13.01.2019) (cit. on p. 4).

[NP] Roberto Navigli and Simone Paolo Ponzetto. *BabelNet: The automatic construction, evaluation and application of a wide-coverage multilingual semantic network*. URL: http://wwwusers.di.uniroma1.it/~navigli/pubs/AIJ_2012_Navigli_Ponzetto.pdf. (accessed: 13.01.2019) (cit. on p. 5).

[PD] Michael J. Paul and Mark Dredze. *You Are What You Tweet: Analyzing Twitter for Public Health*. URL: https://www.aaai.org/ocs/index.php/ICWSM/ICWSM11/paper/viewFile/2880/3264. (accessed: 13.01.2019) (cit. on p. 4).

[Ped+] Fabian Pedregosa, Gael Varoquaux, Alexandre Gramfort, et al. *Scikit-learn: Machine Learning in Python*. URL: http://www.jmlr.org/papers/volume12/pedregosa11a/pedregosa11a.pdf. (accessed: 13.01.2019) (cit. on p. 24).

List of Figures

List of Tables