Computer Science Department
University of Crete

# Opinion Mining on Parliamentary Commentaries, using Machine Learning.

Moschonas Giannis, Smyrnaios Giorgos

Computer Science Department
UNIVERSITY OF CRETE
Heraklion, Greece 2015

Graduate thesis 2015

# Opinion Mining on Parliamentary Commentaries, using Machine Learning.

Moschonas Giannis, Smyrnaios Giorgos

**Computer Science Department**
University of Crete

Computer Science Department
University of Crete
Heraklion, Greece 2015

Opinion Mining on Parliamentary
Commentaries, using
Machine Learning.

Supervisor: Dimitris Plexousakis

Graduate thesis 2015
Department of Computer Science
University of Crete
E-mail (Moschonas): jmoschon@csd.uoc.gr
E-mail (Smyrnaios): gsmyrneo@csd.uoc.gr

Typeset in LaTeX
Heraklion, Greece 2015

Moschonas Giannis, Smyrnaios Giorgos
Computer Science Department
University of Crete

## Abstract

Opinion Mining is the field of Computer Science that studies the association between Natural Language and Computers in an attempt to mine semantically rich information from writing. More precisely, in the wider field of "Natural Language Processing (NLP)", "Machine Translation", "Named Entity Recognition and Disambiguation", "Sentiment Analysis" etc. are included. In this Thesis, a better approach to information extraction from plain texts is attempted. The texts contain commentaries of citizens on consultation of Laws issued by the Greek Government. Extraction of proposals and counter-proposals that the writers of these texts make, as well as of the arguments they formulate, is attempted. Finally, the general opinion of the writer is extracted, which is summarised by a "Positive" or "Negative" characterisation, depending on the writer's view, which comes of the whole text. Mining of the above data is done entirely by analysing the texts through a three-stage analysis (which will be explained in detail in the following chapters of this Thesis) and by utilising techniques that are included in the wider spectrum of NLP (Information Retrieval, Part-of-Speech Tagging, Sentiment Analysis etc.). The results show that we can create realistic methods that can extract this type of information. This matter troubles the Scientific community because these methods can be put into use in a variety of other fields, apart from the field of Computer Science (e.g. Journalism, Politics etc.).

**Keywords:** argument extraction, sentiment, machine learning, suggestion extraction, POS Tagging, opinion mining, natural language processing.

## Declaration of Authorship

We declare that this thesis titled, “Natural Languages Processing on Parliamentary Commentaries, using Machine Learning.” and the work presented in it are our own. We confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where we have consulted the published work of others, this is always clearly at-tributed.
- Where we have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely our own work.
- We have acknowledged all main sources of help.
- Where the thesis is based on work done by ourself jointly with others, we have made clear exactly what was done by others and what we have contributed ourselves.

| | |
|---|---|
| Giannis Moschonas | |
| Name | Sign |

| | |
|---|---|
| Giorgos Smyrnaios | |
| Name | Sign |

## Acknowledgements

We would like to take the opportunity of thanking all these who extended a helping hand whenever we needed one.

We would like to especially thank our advisor, Dimitris Plexousakis, as well as all the members of the Laboratory (Information Systems Laboratory) for the help they provided us with for the completion of this Thesis. More precisely, we would like to thank Irini Fundulaki, Giorgos Flouris, Theodoros Patkos, Panagiotis Papadakos, Katerina Papantoniou, Yannis Roussakis.

But most of all, thank you reader. Having read this line, you at least read one page of our work thesis. Thank you!

Moschonas Giannis, Smyrnaios Giorgos, September 2015

# Contents

# List of Figures

# List of Tables

# 1
# Introduction

The goal of this Thesis is the study and creation of a system which will be able to derive and analyse information from citizens' commentaries on Bills which are available for public consultation from the Greek Government via the OpenGov website (`http://www.opengov.gr/`). More precisely, an effort is made to analyse information that exists mainly in unstructured writing by which citizens can express their opinion on certain law articles of Bills about which the Greek Government has asked for the opinion of the Greek citizens, as well as their separation in suggestions that contain certain information. Also, we attempt to analyse whether the commentaries express a negative or a positive view on the laws or the Bill.

Finally, we extract information about whether a sentence supports the writer's opinion, i.e. whether this sentence constitutes a writer's argument as well as whether the writer suggests something else in the text's sentences. Based on the above, it is considered necessary to set apart three main and autonomous techniques which help in the extraction of this information.

- The first one is "Argument Extraction", which aims at finding the sentences with which the writer argues for or against a law.
- The second is the "Extraction of Suggestions and Counter-Suggestions", which aims at finding the sentences in the text by which the writer makes a suggestion or a counter suggestion.
- The last one is "General Opinion Extraction", i.e. finding out whether the commentator of the law is positive or negative towards it.

## 1.1 Motivation

In the past years, a rapid increase in the information published on the World Wide Web has been noted. It is a fact that recent researches show that the rate keeps increasing very quickly. The phrase that is used in all scientific articles that study the increase of information on the Internet, *"Internet is growing at lighting speed"* is characteristic.

From the above ascertainment we can easily imagine that a big part of the information that is published does not constitute simply a part of a text in a random page. More precisely, it has been noted that the volume of information that is published in Social Media, Social Networks, Forums and generally in public consultation platforms keeps increasing at a rate proportional to the increase of the total information in the World Wide Web.

Therefore a list of questions comes up:

- How can we extract data from unstructured plain texts?
- Is it possible to extract qualitative data (arguments) from these texts?
- Is it possible for the above to be done on every type of texts, without being affected by the subject and the size?

We can think of many more questions.

Based on the above indicative questions, it makes sense to assume that it is necessary for such systems to exist/be improved/be perfectioned. In the research community, the interest for the creation of these systems-tools has existed for years. Nevertheless, we have to highlight that especially during the time that this Thesis is being written, many studies have been conducted and many systems have been created, which help a lot to carry out new studies of this kind (they will be analysed in the next chapter).

Apart from the need that was described in the three questions, it is important to define some practical applications for the issues that are being studied in this Thesis.

But first, we have to give a very brief description of the abilities of the system that was created at the same time with the writing of this Thesis.

First, we have to note that the texts we study are users' answers in the Greek Government's Public Consultation service.

Our system has the following abilities:

- Automatic extraction of proposals and counter proposal that users make in a text.
- Extraction of the argumentation that the writers compose.
- Extraction of the general agreement/disagreement of the writer, imprinted in one word, "Positive" or "Negative".

The above functions, in essence, could help many practical applications, not only in the field of Computer Science by adding the above functions in more complex systems but also in a wide variety of other fields.

- **Search Engines:** So far, many search engines which have the ability to retrieve information that the user is looking for, exist. This search sometimes is done by using methods and algorithms from the field of Information Retrieval (TF-IDF, PageRank etc.) and some cases by using a Query Language if we refer to Link Open Data. In both cases, the way information is retrieved cannot offer more qualitative characteristics, especially if we think about the subject of argumentation. Therefore, let's imagine the abilities that a search engine, which could retrieve information and combine them with key words, would provide us with. In order to make it clearer, we are going to give some examples in query form. *"Find the arguments that are in favour of buying X car". "Find the arguments that contradict the arguments that are in favour of buying X car". "Find all the arguments that are against Bill Y", "Find everyone who is against Bill Z"* etc. [1]

- **Journalism:** The creation of a system which could extract all the above information, would help this field a lot. A journalist could study the argumentation of politicians and draw conclusions about the opinions someone has expressed in a specific amount of time. Of course, we could look for arguments and counterarguments on a matter and also we could check if certain individuals change their stance on a matter as time goes by. We can think of many other scenarios/possible applications.
- **Politics/Administration:** This tool could frame an online consultation service, especially in administration and decision making with the citizens' participation. Some of the functions that the system we develop offers could help in that direction. For example, we could retrieve all the counter proposals/disagreements.

## 1.2 Argument Extraction

First of all, in order to understand this technique, it is necessary to define the term "Argument", as well as the parts that constitute it.

**Argument:** A reasoning with which someone supports or opposes a specific point of view.

The parts of an Argument are "Premise"-"Claim"-"Conclusion". The premise is the sentence with which someone claims/assumes something. Claim is the sentence with which someone tries to reaffirm his assumption. Finally, the conclusion is considered to be the logical implication of the assumption and of the claim. Together with the parts of the argument, it is essential to define the connections between them. [6][7]

- A premise can be a claim and vice versa but not necessarily.
- A premise can oppose a claim or support it.

**Example 1** *"**(1)** Museums and Art galleries offer a better understanding of the Arts to the public, compared to the Internet, **(2)** because in most museums and Art galleries, a detailed description of the artist's background is provided."*

In the above example (1.2.1), we can see an argument that is made up of the assumption **(2)** which is supported by the claim **(1)**.

So, Argument Extraction is defined as the process of identifying arguments together with their components in the text. It is noted that it constitutes a complicated enough process, as the definition of a sentence as an argument with Grammar rules is not clear enough. This is because beyond Grammar rules, the understanding of the text's topic is necessary, as well as the semantics of the words that are directly linked to the topic. There are words whose interpretation changes according to the topic and the style of the text. For example, it is not possible to understand with grammatical analysis (no matter what depths it reaches), whether someone is being ironic or whether a phrase is used in a literal or a figurative sense, let alone in comments which are basically unstructured, brief and with many errors that many times, even if they are made under a specific topic, do not seem entirely relevant.

In this Thesis we will focus on defining a sentence as "argumentative" or "non argumentative". A sentence is considered "argumentative" when it has one of the parts that were mentioned previously. In the previous examples, sentence **(1)** and sentence **(2)** are considered "argumentative" as sentence **(1)** was considered as "claim" and sentence **(2)** as "conclusion".

## 1.3 Suggestion Extraction

In this part,the clarification of the term "suggestion" is considered necessary.

**Suggestion:** It is the reasoning with which someone suggests or makes a counter suggestion about a specific topic.

Defining a sentence as "suggestion" is a clearer Grammatical process because there are specific verb moods that are used to suggest something, for example, the Subjunctive ("to do", "to go") and the Imperative ("do", "eat") but also phrases like "I suggest to...", "It would be nice to...".

**Example 2** *"I suggest lowering the VAT at 10% for all domestic products."*

**Example 3** *"Lower the VAT at 10% for all domestic products."*

In the example (2) the sentence constitutes "suggestion". This is easily distinguished because the verb is in the Subjunctive and there's the expression "I suggest..". The example (3) also constitutes a suggestion because the verb is in the Imperative. It is noted that, as it is concluded from this project, despite the fact that to define a sentence as a "suggestion", Grammar is usually enough, there are cases when someone might come across sentences that meet the Grammar rules we set as prerequisite but they do not constitute "suggestions". There are cases, for example, where a sentence that meets the criteria does not constitute a suggestion but the next sentence, or a close one, might be a suggestion.

**Example 4** *"The changes that must be made are: Vat reduction at 10%. Increase* *in checks for tax evasion."*

In the example (4), the sentence that follows the grammatical patterns for suggestions is the first one but, nevertheless, it is not characterised as a "suggestion". The next ones, though, are suggestions despite the fact that they do not follow the grammatical patterns for suggestions.

## 1.4 Overall Opinion Extraction

By the term "Overall Opinion Extraction", we mean the effort to extract information about the way the writer of the comment expressed his opinion, i.e. whether he is negative or positive towards the topic discussed. This extraction is a difficult task because there are sentences where someone can express a sentiment in an interpretatively controversial way.

**Example 5** *"The weather was nice until it started raining and we didn't go fishing."*

**Example 6** *"The weather was nice until it started raining and unfortunately we didn't go fishing."*

In the example (5), it is not clear whether the writer wrote that sentence with a negative or neutral sentiment, so any interpretation of the way he wrote it is not clear. In contrast, in the (6) example, it is more apparent that the writer has written the sentence to show his displeasure because of the existence of the word "unfortunately".

# 2

# Background

In order to understand, and create the individual parts that were mentioned in Chapter 1 and to interconnect them in total, studying and understanding of both basic and state of the art technologies in the field of Computer Science were essential. Some of the technologies that we studied and used to write this Thesis were the following:

- Machine Learning (2.1)
- POS-Tagging (2.2)
- Information Retrieval (2.3)

Also, a deep understanding of Greek Grammar and of the ways an argument is expressed were necessary.

## 2.1 Machine Learning

Machine Learning is a field of Computer Science that deals with the study and construction of Algorithms that can make predictions about the data that were set as entry. These algorithms function with the construction of a model based on data that exist as entry examples, for prediction and decision making. They are based on these data instead of following strict static instructions. This technology has a wide use in problem solving where a pre-designed algorithm is impossible to use. Sometimes, (like in the case of this Thesis), Machine Learning is confused with Data Mining even though its main focus is the investigative analysis of data. Examples of Machine Learning application are “Spam Filtering”, “Optical Character Recognition (OCR)”, “Search Engines”, ”POS-Tagging” etc.

### 2.1.1 Categorization of Machine Learning Algorithms

The classifications that can be made based on the desired output of the Machine Learning System are the following:

- **Classification:** In this category, the inputs are divided in two or more classes and the system must produce a model that classifies unknown entries in one of these classes. “Spam filtering” is an example of such classification.
- **Regression:** In this category, the outputs are continuous and not distinguishable. This is usually handled in a monitored way.

- **Clustering:** In this category, a total number of inputs must be divided in groups. In contrast with classification, the groups are not known beforehand. For this reason, it can be characterised as an uncontrolled process.
- **Destiny Estimation:** This category finds the allocation of inputs in a place.
- **Dimension Ability Reduction:** This category simplifies the inputs by mapping them out in a lower-dimensional place. An example is when a list of documents is given in natural language and recognising which of these files cover similar topics is needed.

## 2.1.2 Machine Learning Algorithms

Examples of known algorithms which were used in this Thesis follow. In chapter four, the way they were used will be thoroughly analysed.

- **Naive Bayes[1]:** Naive Bayes is an algorithm of probabilistic classification which is based on Bayes Theorem (or Bayes rule), with strongly independent admissions. It is classified in the "Classification" category. The simple admission of the Algorithm is:

$$P\left(C_k|x\right) = \frac{P\left(C_k\right)P\left(x|C_k\right)}{P\left(x\right)} \tag{2.1}$$

  **Example 7** *"To predict the possibility of rain we usually use some indications like the density of dark clouds in the sky."*

$$P\left(raining|darkcloud\right) = \frac{P\left(raining\right)P\left(darkcloud|raining\right)}{P\left(darkcloud\right)} \tag{2.2}$$

  – P(darkcloud/raining) is the possibility of dark clouds when it rains.
  – P(raining) is the possibility of rain in advance.
  – P(darkcloud) is the possibility of a dark cloud in the sky.

- **Support Vector Machines(SVM)[2]:** SVM algorithms monitor learning models connected to learning algorithms which analyse data and recognise templates. They are used for classification and regression analysis. They work taking into account a total of education models which belong to one of the two categories. An SVM education algorithm constructs a model which allocates new examples to one or the other category, making it non-probabilistic binary linear classifier. An SVM model is a representation of the examples as points in space, mapped out in such a way that the examples of the separate

[1]`https://en.wikipedia.org/wiki/Naive_Bayes_classifier`
[2]`https://en.wikipedia.org/wiki/Support_vector_machine`

categories are divided by a clear gap which is as broad as possible. The new examples are then matched to that same space and are predicted to belong to one category based on which side of the gap they fall on. The model of the linear algorithm SVM is:

*Given a train set D, a total of n points of the form:*

$$D = [(x_i, y_i)|x_i \in R^p, y_i \in -1, 1]_{i=1}^{n} \tag{2.3}$$

where $y_i$ is either 1 or -1, which indicates the category to which spot $x_i$ belongs. Each $x_i$ is a $P$-Dimensions "real vector". We want to find the hyperplane of maximum margin which separates the spots which have $y_i = 1$ from those that have $y_i = -1$.

## 2.2 Part-Of-Speech Tagging (POS-Tagging)

In the field of Linguistics, POS-Tagging[3], which is the process of tagging a word in a corpus which matches a specific part of speech, is based on its definition and on its frame, i.e. its relation to other close and relevant words in a phrase, sentence or paragraph. A simplified form is the characterisation of the words as nouns, verbs, adjectives, adverbs etc. POS-Tagging is more difficult than having a list of words and the part of speech each one is, because some words can represent different parts of speech for different periods of time. This is not rare in natural language (in contrast with many artificial languages), where many words have an ambiguous meaning and use. For this reason, the application of Machine Learning is necessary. In more advanced POS-Tagging tools, techniques of Machine Learning are applied, like the Support Vector Machine algorithm that we described in subsection (2.1).

In this Thesis, a ready-made tool for POS-Tagging, "ILSP POS-Tagger" was used, which is based on a dictionary but also uses machine learning algorithms. This tool offers very good precision. More details about it will be given in Chapter 4.

## 2.3 Information Retrieval

Information retrieval(IR)[4] is the scientific field of Computer Science that studies effective ways to search for information and data in files and meta-data relevant to files, as well querying data bases and the World Wide Web. Information Retrieval is based on the data base theory, on suitable informative systems and on mathematical methods of Artificial Intelligence like the ones mentioned in Chapter (2.2), broad use takes place in the World Wide Web and especially by specialists in all search engines (e.g. Google). Someone can say that it has an interdisciplinary character as

[3]`https://en.wikipedia.org/wiki/Part-of-speech_tagging`
[4]`https://en.wikipedia.org/wiki/Information_retrieval`

it borrows elements from Computer Science, Mathematics, Library Science, Cognitive Psychology, Linguistics and Statistics.

A process of information retrieval begins when the user enters a query in the system. The queries are research requests in some data base, like for example the search of an alphanumeric string in some search engine of the Web. In Information Retrieval a query does not precise uni-vocally an item, but may times many items can match the query, maybe in different degrees of relevance.

An item is an entity that is represented by some information in a data base. In order to answer the user's queries, research among the data of the base for possible answers is done. The data can be text documents, pictures, sounds or video archives.

Most information retrieval systems calculate a numerical score about how well each item in the data base matches the query and classify the items according to this calculation. The items that are on top of the list appear in front of the user. The process can be repeated afterwards, if the user wishes to improve the query.

### 2.3.1 Information Retrieval Algorithms

Example of a well-known algorithm follow, which were also used in this Thesis. In Chapter 4, the way they were used will be analysed in detail.

- **TF-IDF (Term Frequency-Inverse Document Frequency):** The TF-IDF algorithm is an arithmetic statistical element whose goal is to reflect how important a word in a file or in a collection, is. Often, it is used as a weighting factor in information and text retrieval. The value of TF-IDF increases in proportion to the number of times that a word appears in a text, but is counterbalanced by the frequency of the word that exists in the collection, which helps to adapt to the fact that some words appear, in general, more frequently.

  To interpret TF-IDF it is necessary to note that it is a product of two algorithms (term frequency, inverse document term frequency) which must be explained.

  The term frequency $tf_{t'd}$ of the term t in a file is defined as the number of times that $t$ appears in $d$.

  $$tf\,(t,d) = 0.5 + \frac{0.5 * f\,(t,d)}{max\,\{(t,d) : t \in d\}} \tag{2.4}$$

  Inverse document frequency is a way to measure how much information each word contains, i.e. if the term is frequent or rare in all the files. It is the logarithmically weighted fraction of the files that contain the word. It comes up from the division of the total number of files from the number of files that contain the term. In that way, we define the IDF (Inverse Document Frequency) of t as:

$$idf\,(t, D) = log * \frac{N}{\mid \{d \in D : t \in d\} \mid} \tag{2.5}$$

Where:

- $N$ is the total number of the collection's texts.
- $\mid \{d \in D : t \in d\} \mid$ the number of the texts that the term appears in.So,as TF-IDF is defined the product of $tf$ and $idf$.

## 2.4 Related Work in Argument extraction

One of the first approaches for argument identification is presented in the Doctoral Thesis entitled "Argumentative Zoning", written by Teufel (1999). After this Thesis, there have been a variety of projects and Theses that are focused on how to use the public's opinions that are available in texts of electronic form, aiming at defining social tendencies and analysing their political stance. For example, the WeGov toolbox[1] which is an online tool (http://wegov-project.eu) that gives someone the possibility to search for topics and views from different social networks. It supports the analysis and the brief presentation of views and discussions and ultimately helps those in charge of Politics to publish the information that has been extracted from the Social Media. The system even predicts stances that are expected to attract more attention.

An application focused on Sentiment Analysis in political comments on Twitter is the Thesis of Diakopoulos and Ahamma[2] Diakopoulos and Shamma,2010) which associate timelessly the dynamics of a sentence in a Twitter discussion. For this goal, the writers use a commentary process based on Mechanical Turk (which is supported by many filters) to note the Tweets that are relevant to the discussion. In the Thesis, the total sentiment of Tweets was studied. Another study from Tumasjan et al. [3] verifies that micro blogging (Twitter) is widely used for political discussion and expression of political view. In this Thesis,to achieve Sentiment Analysis, the writers used LIWC2007 which is a tool for Language Analysis.

One more Thesis which attempts to extract arguments and separate them in Claim and Justification in texts in Greek found in Social Media is "Argument Extraction from News,Blogs and Social Media"[4] and achieves a total precision by 72% and recall by 79%.

Finally, this Thesis was based a lot on a paper of [7] Christian Stab and Iryna Gurevych which is divided in two basic steps. First of all, in tracking the elements of the argument which are either "justification" or "claim" and secondly to define pairs from these components which together form a complete Argument. In their experiment, the results were F1-Score 72.6% for tracking the argument's components and 72,2% for tracking the connections that constitute a complete argument.

# 3
# Methods

In this chapter, we will thoroughly analyse the ways with which the three processing stages which were presented in the previous units, were implemented.

In order to describe in the best possible way the process that was followed, a detailed description of the dataset which was used will be given, as well as of the features that characterise it. Then, we will describe the relational database model which we used to store the information from the texts (dataset). Finally, for each one of the process stages, we will describe the methodology with which each matter was approached.

## 3.1 Preparing the Dataset

In this part of the methodology, the features of the used dataset (3.1.1) will be described in detail. Some information on the way of choosing data (3.1.2) will be described as well. Next, the way of data mining from the Greek Open Government platform[1] (3.1.3) and finally, the Entity-Relation Model (3.1.4) of the database which was used to store the data will also be described.

### 3.1.1 Dataset

As it has already been mentioned in some points of this Thesis, the data that were used have been taken from the Greek Open Government platform, which constitutes a platform of electronic consultation of citizens on texts, more specifically on laws and decrees that the Greek Government issues. These data are open and accessible to everyone.

In this section, the basic features of the studied texts will be described. The reason why this section comes first in this part of the methodology, is that the very nature of these texts (they are basically users' comments to the online service), created many problems in their analysis.

As it has already been mentioned, the texts that were studied feature several oddities, some of which made the process of analysing them difficult.

[1] `http://www.opengov.gr`

- Initially, the first that we can notice is that the length of the texts is relatively short. To be precise, it is rare for them to be longer than 3000 characters (approximately 200 words, 80% of the texts). The length of the text did not affect all the stages of the analysis. The biggest difficulty appeared in the effort to extract the degree of the writer's agreement with the initial text (more details will be given later).

- A second remark is the fact that the texts that were studied are not consist an official text. By the term "official", we want first to declare that the texts are made up of users' comments in an online service and secondly, that they contain many errors (spelling etc). This created many difficulties in the studying of these comments. The first difficulty had to do with the tools needed in order to conduct the overall analysis of each text. The basic idea was that the tools had to be tolerant when it came to errors, at least up to a degree.

  Some very usual errors are:
  1. spelling errors
  2. absence of some letters in a word
  3. letter transposition in a word
  4. use only of capital letters
  5. absence of punctuation
  6. wrong sentence separation (there was no gap after the dot)
  7. other types of errors

- One more issue is that there are many times when syntactical structure errors are spotted. This problem is directly connected to the use of POS-Tagger for the syntactic analysis (parsing) of texts. This issue affects, to some degree, the extrapolation of arguments and of proposals and counter proposals that the user makes.

- Another feature is that the texts are entirely in Greek. This problem is more serious, because there are no tools which we needed at some point of the analysis, that support the Greek Language. Subsequently, as we will see later on, there was the need to resort to some compromising solutions.

- One last issue that is worth mentioning, which constitutes a more qualitative feature, at least in the whole of texts that were studied, is the fact that the majority of users who wrote a comment are "annoyed". This "annoyance" stems from the fact that the texts that are under discussion contain laws and presidential decrees that, essentially, lead to a decrease in public spending towards the citizens. This "annoyance" is noted almost in the entire dataset that we studied. The problem is that the texts in which the writers agree with the initial text are limited. As a result, this issue complicates the process of detecting, if the writer agrees with the initial text. Also another serious problem is that the "annoyance" causes the users to make several kind of mistakes (such as grammatical, syntactical etc.).

### 3.1.2 Choosing the Set of Documents

To continue the process, we randomly chose five different bills that contained a significant amount of users' replies. Afterwards, we selected a few, trying to eliminate the replies that we did not want to process. For example:

- replies that only contained one sentence
- replies in greeklish

Next, we limited the dataset so as to contain a number of approximately two hundred replies. This total is the final dataset that was studied and on which the conclusions for all the processing stages were based.

### 3.1.3 Finalizing the Dataset

The last step for the creation of the final dataset was the data mining from the Greek Open Government platform[2] (the website for public consultation on laws). This process was simple enough, since the service provides the users with the option to locally store all the comments that have been posted for each law or decree. The data were in excel file format, providing for each comment the following meta-data:

- the Law Article which was commented
- an id for each comment
- the name of the user-commenter
- the date

These data were later stored in the database which was created for the storage of data that were collected in all the stages of processing.

### 3.1.4 Database

In order to store the database, an MySQL[3] Database was created, whose Entity-Relation Model[4] can be presented in the following layout.

[2]`http://www.opengov.gr`
[3]`https://www.mysql.com`
[4]`https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model`

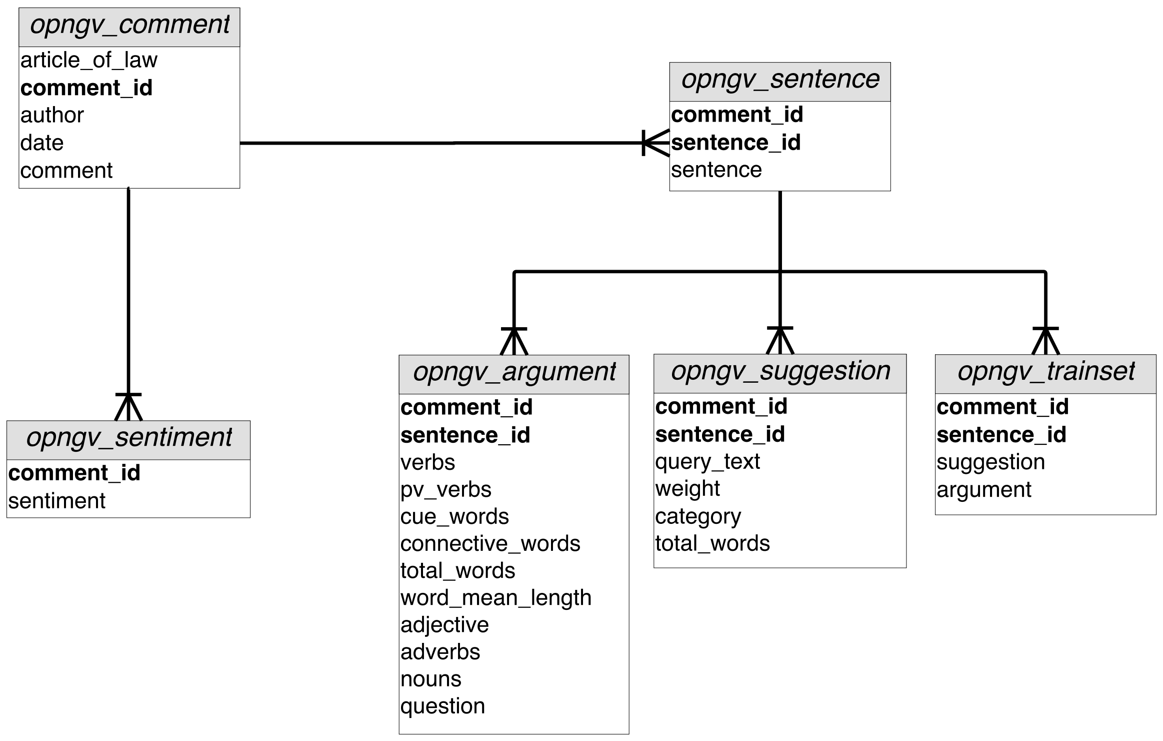


**Figure 3.1:** Entity - Relation Model

In the above layout, we can see the Entity-Relation Model that was used for storing data. Next, the tables which appeared in the above diagram will be described in detail:

- **opngv_comment:** It is used to store texts which have been extracted from the public consultation web page and more precisely,each user's texts is stored in the variable of the "comment" table. The rest of the elements of this text are used to store some meta data of the texts (e.g. law name/number, name of the author, date).
- **opngv_sentiment:** This table is used to store the information that is extracted after the completion of the process. More precisely, in the "Sentiment" field, the information that is stored defines whether a user's text/comment is negative or positive,
- **opngv_sentence:** This table is used to store sentences that make up the text. In the "sentence" field, each sentence is stored. The separation of the text is done by using POS-Tagger.

- **opngv_argument:** It contains the values of the argument markers for each sentence. Each variable of this table is explained in unit (3.2.1).
- **opngv_suggestions:** Respectively with the “opngvargument” field, this field contains the suggestion markers.
- **opngv_trainset:** This table contains data which declare whether a sentence is an Argument or a Suggestion. These data are stored in the boolean variables of the table (argument, suggestion). The data of this table make up the train set that is used for machine learning. More precisely, the train set comes up by combining the “opngv_argument” and the “opngv_suggestion” tables with this table.

In all the table,the primary keys have been marked with bold.

### 3.1.5 Building a Trainset

One last element that deals with the chapter on dataset, is characterising a total of sentences if each one of them contains an Argument or a Suggestion. We should note here that sentence separation will be analysed thoroughly later (3.2.2.1). The Train set that we created, as we will see in the chapters that follow, is needed so that the machine learning algorithms can be trained, as well as to achieve a better evaluation. The set of sentences that was created contains approximately one thousand sentences.

## 3.2 Argument Extraction

At this point, the process with which the Argument Extraction was carried out on the whole of the texts will be described. The process is based on three stages:

- Selecting Argument Markers (3.2.1).
- POS-Tagging the set of documents (3.2.2).
- Applying machine learning to the Argument Markers (3.2.3).

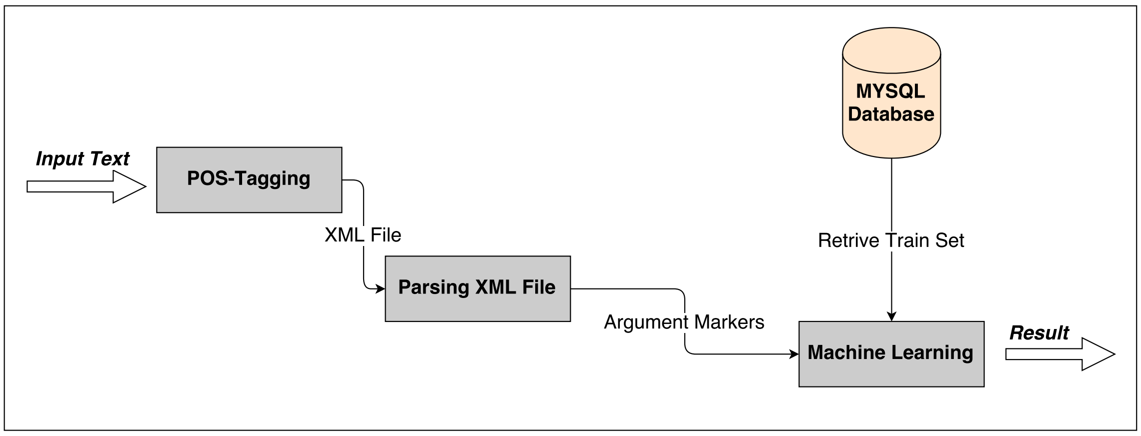


**Figure 3.2:** Argument Extraction Process

Each one of the three stages will be explained right away.

### 3.2.1 Selecting Argument Markers

This step, in essence, constitutes the selection process of certain "criteria" - "Argument Markers" that will help us define what an argument is. Essentially, this process constituted the hardest part in the whole Argument Extraction stage.

For this step, we decided to use Machine Learning techniques, an approach that is popular in the related literature (3.2.1].In particular, our approach consists in defining some parameters which will define will define with a relative clarity whether a sentence is an argument or not.

But before we get in the process of searching for these variables, we had one more difficulty to face. Even though it sounds relatively easy to recognise the arguments in a text, in reality it was a rather difficult process. In experiments with human users, it turned out that there was great difficulty, even among humans, to agree on whether a sentence contains an argument or not. To address this problem, we formulated the following definition:

*Sentence is likely to contain an argument if it contains the following markers:*

- *The sentence provides a context clue from which we can interpret that the writer expresses an opinion.*
- *The sentence is explanatory, which means that the writer wants either to further explain or support an opinion.*

If there is any of the above markers in a sentence, then this sentence can be characterised as an argument (Argumentative sentence). Even with the above two conventions, in some cases it is still difficult to determine whether a sentence is an argument. More precisely, another convention has been set, with which we made an effort to exclude the sentences that are interrogative. The reason is that usually, interrogative sentences are likely to express sarcasm. In our dataset this was actually very usual.

Having set the above conventions, we studied a total of Argumentative Sentences, in order to be able to track down “Argument Markers”. In essence, we looked for parts of speech (for example the number of verbs, number of adjectives and more), as well as other variables that often appear in this type of sentences. These variables were to be used in the next steps and especially in the step where the “Machine Learning” process is applied

From the study that was conducted, also combining Related work, we ended up with the following variables:

- **Number of Verbs:** In many studies which have been conducted in the past, it was noted that verbs are closely linked to arguments. To be precise, verbs express action. Another characteristic is that verbs syntactically compose a sentence or, a little more arbitrarily, verbs enrich a sentence. Consequently, it was noted that the sentences which contain arguments usually have a larger number of verbs.
- **Number of Verbs in Passive Voice:** The bibliography that we had available mentions that usually the verbs that are in the passive voice are used when someone claims something concerning a particular matter. In our dataset this was not so clear. However, in the measurements done, we noted that most sentences containing an argument also contain, to a great extent, verbs that are in the passive voice.
- **Key Words:** This variable refers to the number of words that were traced in a group of words that we created. This group contains words that are used when someone tries to explain or support an opinion. For example, in this group there are the words consider, believe, admit, suppose, think, must, consequently, since, until, because, namely.
- **Number of Connective Words:** This variable counts the number of linking words that were found in a sentence. In essence, this number has a direct relation to the next variable which shows the total number of words. The idea of counting the length of a sentence lies to the fact that usually, longer sentences are more likely to contain arguments. Especially when the studied dataset contains texts with arguments and political discussion.
- **Total Number of Words:** similar to the previous variable.
- **Average Number of Letters in a Word:** This variable came up from the bibliography that we had available. It has been noted that this variable, especially in texts that contain political discussion, can help in a significant way to track arguments.
- **Number of Adjectives:** It is the number of arguments that were found in a sentence. The logic behind this variable is the same with the “Number of Verbs” variable logic.
- **Number of Adverbs:** It came up after studying the reference and it states the number of adverbs in a sentence.

- **Number of Nouns:** The role of this variable is equivalent to the "Number of Adjectives" variable.
- **A Boolean Variable that States Whether a Sentence is Interrogative:** The need for this variable was explained thoroughly above.

### 3.2.2 POS-Tagging

In this part we will analyse the second step in the process of argument extraction from a text. Unlike the previous step (3.2.1), the execution of this step is essential for every new step we wish to analyse.

As it has already been mentioned above (2.2), the POS-Tagging procedure has as goal to analyse the grammar of the text, as well as to extract an output which will contain a recognition of what part of speech each word is. Clearly, this process is very important for Argument Extraction because it constitutes the way with which all parts of speech will be detected.

#### 3.2.2.1 POS-Tagger

In this Thesis, "ILSP POS-Tagger[5]" was used, which was created by the "Institute of Language and Speech processing[6]". Also the POS-Tagger is used in order to split the text into sentences. This feature appears in the next section (3.2.2.2) in which we can see the output that the Tagger produces.

#### 3.2.2.2 POS-Tagger Output

From the outputs supported by POS-Tagger that we had available, we have chosen the "xceslemma" option. With this option beyond POS-Tagging, Lemmatization can also be accomplished. We will need the latter in the next unit that we are going to study. The output given is in XML format. An example can be seen below:

[5] `http://nlp.ilsp.gr/soaplab2-axis/`
[6] `http://www.ilsp.gr/en`

```
<?xml version='1.0' encoding='UTF−8'?>
<cesDoc xmlns="http://www.xces.org/schema/2003" version="0.4">
  <text>
    <body>
      <p id="p1">
        <s id="s1">
          <t id="t1" word="..." tag="AtDfNeSgNm" lemma="..."/>
          <t id="t2" word="..." tag="RgFwOr" lemma="..."/>
          <t id="t3" word="..." tag="PnReNe03SgNmXx" lemma="..."/>
          <t id="t4" word="..." tag="VbMnIdPr03SgXxIpPvXx" lemma="..."/>
          <t id="t5" word="..." tag="VbMnIdPr03SgXxIpPvXx" lemma="..."/>
          <t id="t6" word="..." tag="AsPpSp" lemma="..."/>
          <t id="t7" word="..." tag="NoCmFeSgAc" lemma="..."/>
          <t id="t8" word="..." tag="RgFwOr" lemma="..."/>
          <t id="t9" word="..." tag="PTERM_P" lemma="..."/>
        </s>
      </p>
    </body>
  </text>
</cesDoc>
```

The text that was given as input was *"The output which is given is in XML format"*. We note that for every word of the text, the following information is given:

- **Id:** an auto increment identifier given in every word of the text
- **Word:** the initial word of the text
- **Tag:** the Grammar concerning the particular word. For example, verbs start with "Vb" and nouns with "No". The whole tagset can be found here: `http://nlp.ilsp.gr/nlp/tagset_examples/tagset_en/`.
- **Lemma:** it is the dictionary entry of each word. For example, we note that the verb "given" has "give" as its lemma.

Although we gave a link to the tagset of the POS-Tagger, it is important at this point to give a brief explanation. For example, the tag "VbIsIdPr03SgXxIpAvXx" means:

- **Vb:** verb
- **Is:** Impersonal
- **Id:** Indicative
- **Pr:** Present
- **03:** Third Person
- **Sg:** Singular
- **Xx:** Gender, No Value
- **Ip:** Imperfective
- **Av:** Active
- **Xx:** Case, No Value

#### 3.2.2.3 Parsing XML File

For the operation and data extraction from the XML file, DOM (Document Object Model[7]) was used, which is a platform for the representation and the interaction with objects in XML files. In this platform, the nodes of each file are organised in a tree form called the DOM tree. The objects in the DOM tree can be operated using the methods for objects. DOM's public interface is determined in Application Programming Interface.

#### 3.2.2.4 Uploading to Database

Finally, the data from the Parsing of XML file (3.2.2.3) are stored in the Database and more precisely in the "opngv_argument" table. For storing, the following Query is used:

```
INSERT INTO opngv_argument VALUES (values..)
```

### 3.2.3 Apply Machine Learning

The last stage for argument retrieval consists of the application of a Machine Learning process. In order for this process to be accomplished, a train set and a test set should have been clearly set.

#### 3.2.3.1 Selecting Train and Test Set

In this case in point, the train set is stored in the Data Base. More precisely, the train set retrieval from the Database is done with the following query:

[7] `http://www.w3.org/DOM/`

```
SELECT
        opngv_argument.verbs,
        opngv_argument.pv_verbs,
        opngv_argument.cue_words,
        opngv_argument.connective_words,
        opngv_argument.total_words,
        opngv_argument.word_mean_length,
        opngv_argument.adjective,
        opngv_argument.adverbs,
        opngv_argument.noons,
        opngv_argument.question,
        opngv_trainset.Argument
FROM
        opngv_sentence
        INNER JOIN opngv_argument
                ON opngv_sentence.comment_id = opngv_argument.comment_id
                AND opngv_sentence.sentence_id = opngv_argument.sentence_id
        INNER JOIN opngv_trainset
                ON opngv_sentence.comment_id = opngv_trainset.comment_id
                AND opngv_sentence.sentence_id = opngv_trainset.sentence_id
```

On the other hand the test set constitutes the total of the sentences of a text, which we wish to analyse. In essence, to make it clearer, two tables, which contain the same columns, are created every time. The train set table contains the data as described in units (3.1.5) and (3.2.1). The second table (test set), contains the data that came up from the process that was described in unit (3.2.2) for new texts.

#### 3.2.3.2 Machine Learning Process

For the application of a Machine Learning Process, the Weka Framework "Weka 3: Data Mining Software in Java[8]" was used.

*"Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a dataset or called from your own Java code. Weka contains tools for data pre-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes."*

#### 3.2.3.3 Machine Learning Algorithms

To verify the accuracy of the Train set, we tried many algorithms:

- Logistic Regression[9]
- Native Bayes[10]

[8] `http://www.cs.waikato.ac.nz/ml/weka/`
[9] `https://en.wikipedia.org/wiki/Logistic_regression`
[10] `https://en.wikipedia.org/wiki/Naive_Bayes_classifier`

- Support Vector Machines[11]
- Random Forest[12]

The results will be analysed in the following units.

## 3.3 Suggestion Extraction

At this section we will describe the procedure that we followed in order to achieve the Suggestion Extraction. The Procedure has five steps:

- Selecting Suggestion Markers (3.3.1)
- POS-Tagging & Lemmatization the set of Documents (3.3.2)
- Apply Information Retrieval Methods in order to find the suggestions (3.3.3)
- Adding additional features for the optimization of Machine Learning Process (3.3.4)
- Apply Machine Learning (3.3.5)

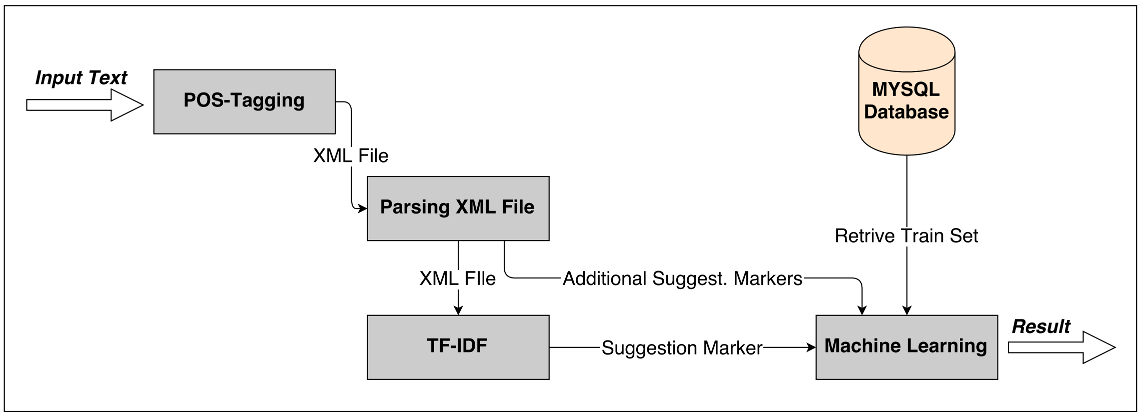


**Figure 3.3:** Suggestion Extraction Process

Each one of the above steps will be explained in detail, in the next sections.

### 3.3.1 Selecting Suggestion Markers

As in the previous chapter we have, at this stage, to set some suggestion markers i.e., some characteristics in each sentence, which, if they appear, then the sentence that is being studied probably contains a suggestion.

The basic idea that we applied is based, to a great extent, on Greek Grammar. We found that a suggestion can be "modelled" better than an argument. The basic concept is that in order to make a suggestion, a writer is "obliged" to follow some Grammar rules.

[11]https://en.wikipedia.org/wiki/Support_vector_machine
[12]https://en.wikipedia.org/wiki/Random_forest

We know that according to Greek Grammar, in order to express incitement in a sentence, the writer should use mainly either the Imperative or the Subjunctive Grammatical mood. These two Grammatical moods are often linked to the meaning we want to detect in our texts.

Starting with the aforementioned concept, we noted that the use exclusively of Greek Grammar is not enough. In some way, we had to specify with more clarity what we were looking for in a text.

**So, we extended the above concept. We combined Grammar rules with lemmas in order to achieve higher accuracy.**

To make the above process clearer: Initially, using the "ILSP POS-Tagger" output, we noted the Grammar tags (3.2.2.2), as well as the lemmas of verbs (3.2.2.2) in order to create a list of verbs that are involved in cases where a sentence can be considered as a suggestion. We can say that we noted the verbs which define the meaning of a sentence, making it a suggestion.

An example can make the concept that was presented even clearer: Given the sentence *"a change in paragraph 5 of the law must be made.."* we noted that, in this example, "must be made" plays a determining role in order to put across the meaning of the sentence as a suggestion. This particular part of the sentence also has a particular Grammatical form.

```xml
<?xml version='1.0' encoding='UTF−8'?>
<cesDoc xmlns="http://www.xces.org/schema/2003" version="0.4">
  <text>
    <body>
      <p id="p1">
        <s id="s1" casing="lowercase">
          <t id="t1" word="..." tag="VbIsIdPr03SgXxIpAvXx" lemma="..."/>
          <t id="t2" word="..." tag="PtSj" lemma="..."/>
          <t id="t3" word="..." tag="VbMnIdXx03SgXxPePvXx" lemma="..."/>
          <t id="t4" word="..." tag="NoCmFeSgNm" lemma="..."/>
          <t id="t5" word="..." tag="AsPpPaFeSgAc" lemma="..."/>
          <t id="t6" word="..." tag="NoCmFeSgAc" lemma="..."/>
          <t id="t7" word="..." tag="DIG" lemma="..."/>
          <t id="t8" word="..." tag="AtDfMaSgGe" lemma="..."/>
          <t id="t9" word="..." tag="NoCmMaSgGe" lemma="..."/>
          <t id="t10" word="..." tag="PTERM_P" lemma="..."/>
        </s>
      </p>
    </body>
  </text>
</cesDoc>
```

More precisely, we noticed that some tags, such as “VbLsLdPr03SgXxLpAvXx” and “VbMnLdXx03SgXxPePvXx” appear in many cases. Finally, we created a list with many cases that we detected. Some cases appear below:

- We have to VbLsLdPr03SgXxpAvXx PtSj
- PtSj VbMnLdXxPepvXx must be studied
- VbMnLdXx03SgXxPePvXx PtSj must be applied
- Add PtSj VbMnLdPr03SXxLpPvxX
- ...

The above list is, in essence, the suggestion markers that we created. (The full list contains about eighty suggestion markers. The list is part of the submitted code.) Also some information about the POS-Tagger tagset can be found in section (3.2.2.2).

### 3.3.2 POS-Tagging and Lemmatization of the set of Documents

The second stage for suggestion mining lies in the POS-Tagger performance. This process is the same as the one described in unit (3.2.2) and for the sake of brevity it will not be analysed again here.

### 3.3.3 Apply Information Retrieval Methods in order to find the Suggestions

Having created the suggestion markers, as we saw in subsection (3.2.1), we have, at this point, to analyse the way with which we can find similarities between the sentences being analysed and the suggestion markers.

Initially, for every sentence in the process of analysis, the POS-Tagging & Lemmatization method (3.3.2) is applied. From the Output of this process we make good use of the information that is absolutely essential for this step. So, for every sentence we keep the following information:

- the verb tag.
- the verb lemma.
- the lemmas of words “to”, “will”.
- tags of the words “to”, “will”.
- “Qst” if the sentence is interrogative.

So, the sentence that we saw above, *“a change must be made”* in paragraph 5 of the law” becomes *“VbLsLdPr03SgXxLpAvXx should PtSj be VbMnLdXx03SgXxPePvXx made”*.

The next step is to create a similarity score between the sentence that came up and each one of the suggestion markers. The comparison was made using TF-IDF[13] and Jaccard Similarity[14] and as we will see in the following subsection, the results were satisfying enough.

One more point that we need to stress is that we divided the suggestion markers in 4 categories, depending on the Grammatical mood, "Imperative" or "Subjunctive" and depending on whether the verbs are in a Past Tense (should have). The categories, as well as all the suggestion markers can be found in the "suggestionQ" folder through the online repository[15].

Finally, having made the above comparisons, only the highest score will be kept, as well as the category from which this score came up. We should also note that the sentences that contain "Qst" automatically get zero score because we wanted to exclude the interrogative sentences for the same reasons that were mentioned earlier.

### 3.3.4 Adding additional features for the optimization of Machine Learning Processs

In order to achieve better results in each sentence that resulted from the above process, we counted the number of words from which it is made up.

### 3.3.5 Apply Machine Learning

The last stage for the retrieval of suggestions is the application of a Machine Learning Process. To implement this process, a train set and a test set must be definitely determined.

#### 3.3.5.1 Selecting Train and Test Set

In this case the train set is stored in the Database. To be precise, the retrieval of the train set from the Database is done with the following query:

[13] `https://en.wikipedia.org/wiki/Tf%E2%80%93idf`
[14] `https://en.wikipedia.org/wiki/Jaccard_index`
[15] `https://github.com/csdRepo/OpinionMining.git`

```
SELECT
        opngv_suggestion.weight,
        opngv_suggestion.category,
        opngv_suggestion.total_words,
        opngv_trainset.Suggestion
FROM
        opngv_sentence
        INNER JOIN opngv_suggestion
                ON opngv_sentence.comment_id = opngv_suggestion.comment_id
                AND opngv_sentence.sentence_id = opngv_suggestion.sentence_id
        INNER JOIN opngv_trainset
                ON opngv_sentence.comment_id = opngv_trainset.comment_id
                AND opngv_sentence.sentence_id = opngv_trainset.sentence_id
ORDER BY
        opngv_trainset.Suggestion DESC
LIMIT 366
```

In this case, too, the test set consists of the sentences of the text, more precisely, the sentences that came up from the process (3.3.3).

#### 3.3.5.2 Machine Learning Process

As in the Argument Extraction (3.2.3.2), the Weka Framework was also used here. Again, for the sake of brevity, it will not be analysed here.

#### 3.3.5.3 Machine Learning Algorithms

To verify the accuracy of the Train Set,we tested enough Algorithms:

- Native Bayes[16]
- Support Vector Machines[17]
- Random Forest[18]
- J48[19]

## 3.4 Overall Opinion Extraction

In this final chapter, the third leg of this Thesis, where the possibility of recognising the general degree of the writer's agreement with regards to the initial public consultation text of the Greek Government was studied, will be analysed. For this process, no kind of meta-data which would have supplied this information automatically (Agree/Disagree button, Like etc.), was used. The whole process has been completed by studying exclusively the users' texts.

[16]https://en.wikipedia.org/wiki/Naive_Bayes_classifier
[17]https://en.wikipedia.org/wiki/Support_vector_machine
[18]https://en.wikipedia.org/wiki/Random_forest
[19]https://en.wikipedia.org/wiki/J48

For the effective mining of this kind of information from the texts, we used the "Sentiment Analysis" method, in an attempt to recognise the sentiments with which the users write their texts. Having analysed the sentiments, we noted that very often it seems that they match each writer's degree of agreement. In more detail, this process can be divided as follows:

- Translate Documents to English (3.4.1).
- Perform Sentiment Analysis (3.4.2).

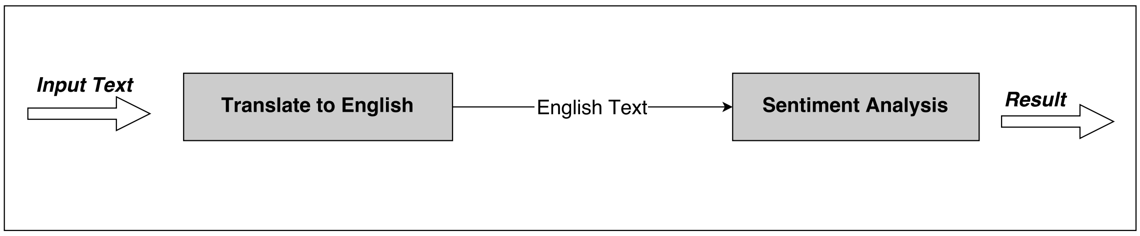


**Figure 3.4:** Suggestion Extraction Process

### 3.4.1 Translate Documents to English

In order to be able to perform Sentiment Analysis, we had, in the first stage, to automatically translate the texts in the English language. The reason is that all the tools that are available do not support the Greek language. Also, studying and developing software that could perform Sentiment Analysis for Greek could constitute an independent task.

It makes sense to think that with automatic translation, there are many times where the meaning of a sentence is altered and the translated text has many mistakes. This problem is even greater in Greek, because it has a complex and rich Grammar and subsequently, the texts in this case may be even more altered.

Nevertheless, the above alterations do not affect the application of Sentiment Analysis. This happens because during Sentiment Analysis implementation, we are more interested in the connotation of the words, i.e. how much of a negative or positive meaning a word gives to a text.

For the automatic translation of texts from Greek to English, "Google Translate" has been used, a service provided for free through the appropriate API[20].

### 3.4.2 Perform Sentiment Analysis

In the demo that accompanies this Thesis, there is the possibility to use two external tools that provide this kind of option.

[20]https://cloud.google.com/translate/

- **SentiStrength[21]:** *"SentiStrength estimates the strength of positive and negative sentiment in short texts, even for informal language. It has human-level accuracy for short social web texts in English, except political texts. SentiStrength reports two sentiment strengths:*

  - *-1 (not negative) to -5 (extremely negative)*
  - *1 (not positive) to 5 (extremely positive)*

  *Why does it use two scores? Because research from psychology has revealed that we process positive and negative sentiment in parallel - hence mixed emotions. SentiStrength can also report binary (positive/negative), trinary (positive/negative/neutral) and single scale (-4 to +4) results."*

- **Sentiment Analysis with Python NLTK Text Classification[22]:** *"Sentiment analysis using a NLTK 2.0.4 powered text classification process. It can tell you whether it thinks the text you enter below expresses positive sentiment, negative sentiment, or if it's neutral. Using hierarchical classification, neutrality is determined first, and sentiment polarity is determined second, but only if the text is not neutral."*

[21] `http://sentistrength.wlv.ac.uk`
[22] `http://text-processing.com/docs/sentiment.html`

# 4 Evaluation and Results

In this chapter, the efficiency of the mechanisms that were studied above will be analysed, on whether they can achieve targeted data extraction. The experiments that were carried out, as well as the results that will be presented below are divided in three units, depending on the kind of data we want to extract:

- Argument Extraction (4.1)
- Suggestion Extraction (4.2)

## 4.1 Argument Extraction

In this section the results of the methods that were introduced in the previous chapters on the writer's argument extraction will be presented. Especially, we are going to try to present some indicators for the Argument Markers that we set. These indicators show the "ability" of each variable to "find" an Argument (4.1.1). Also, we are going to see the efficiency of each Algorithm that we tried in this procedure (4.1.2). Finally, we are going to present some diagrams that show the efficiency of the Train Set that we have created on the writer's Argument Extraction from the texts (4.1.3).

### 4.1.1 Argument Markers

As we saw above, the corpus that was studied had a total of 1015 sentences, which make up approximately two hundred users' texts (user commentaries). Initially, we need to stress that in the corpus, the number of sentences that were characterised as Argumentative is relatively bigger than the number of sentences that were not characterised as Argumentative. This property is shown in the following diagram.

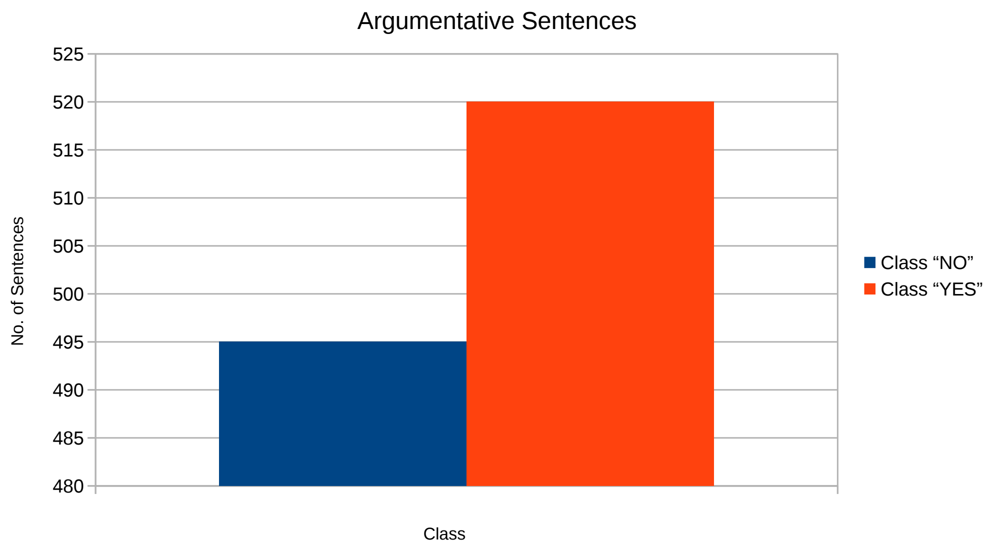


**Figure 4.1:** Argumentative Sentences in Train Set.

In the rest of this unit (4.2.1), class "yes", i.e. the sentences that are "Argumentative" will be symbolised with red and respectively, the sentences belonging to class "no" will be symbolised with blue.

In the next diagrams we can see for each one of the Argument Markers that we have set, the ability of each variable to define an Argument. The axis X shows the frequency of each variable in each sentence. What we note in most diagrams, is that as frequency increases, class "yes" surpasses class "no".

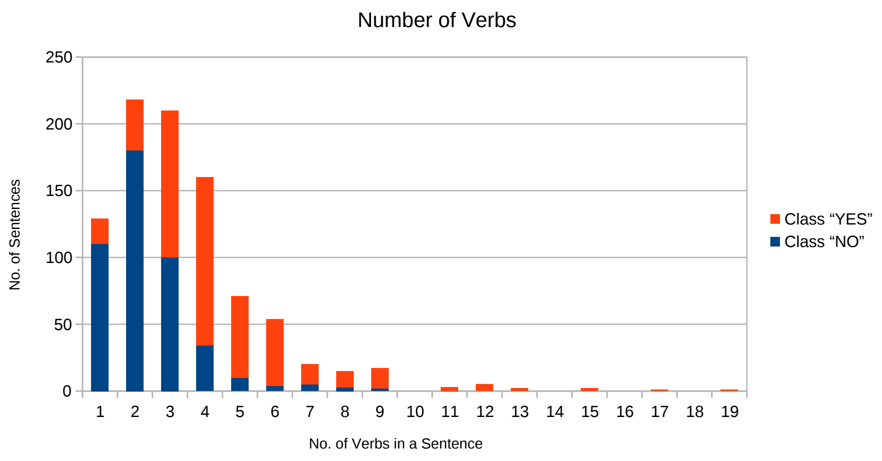


**Figure 4.2:** Argument Marker - Number of Verbs in a Sentence.

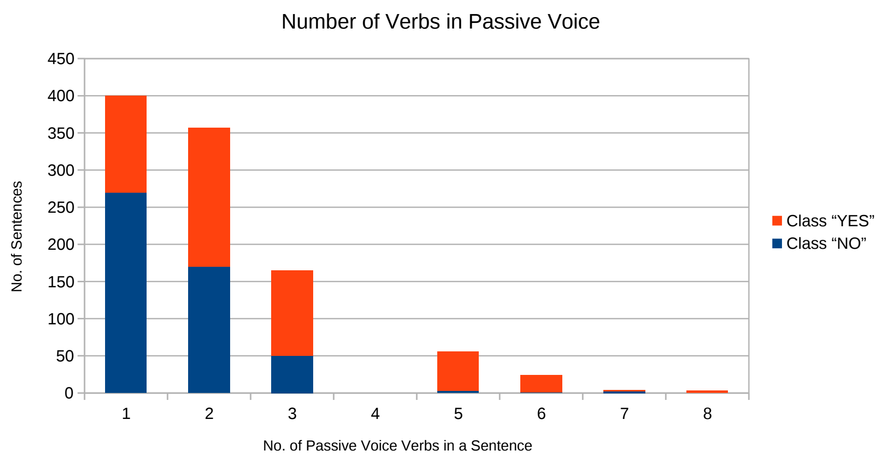


**Figure 4.3:** Argument Marker - Number of Verbs in Passive Voice in a Sentence.

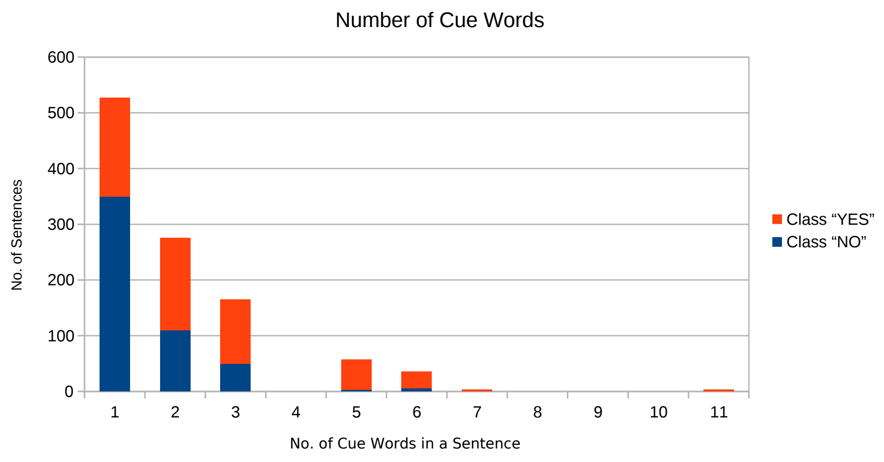


**Figure 4.4:** Argument Marker - Number of Cue Words in a Sentence.

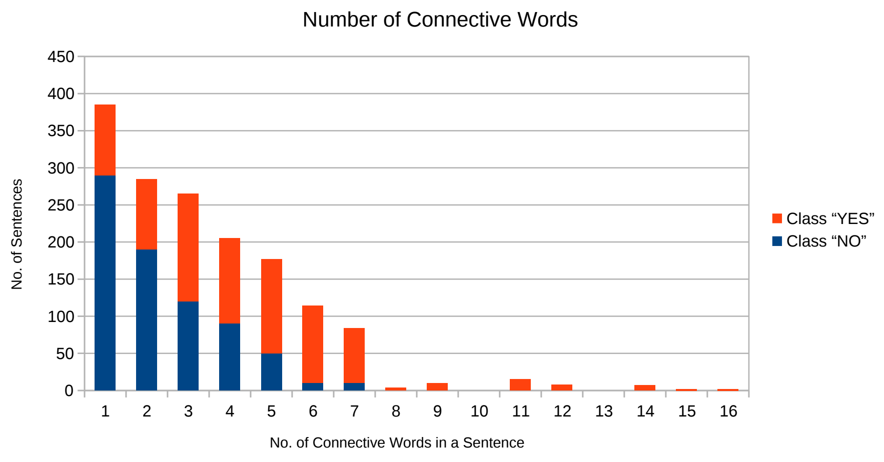


**Figure 4.5:** Argument Marker - Number of Connective Words in a Sentence.

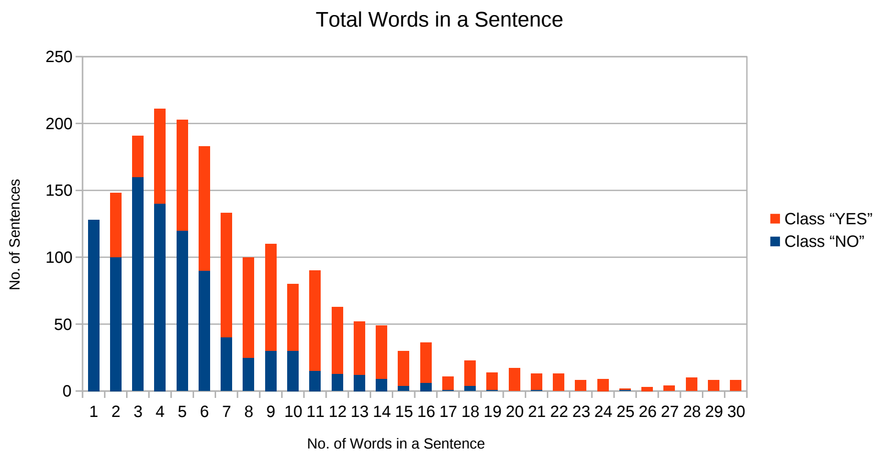


**Figure 4.6:** Argument Marker - Total words in a Sentence.

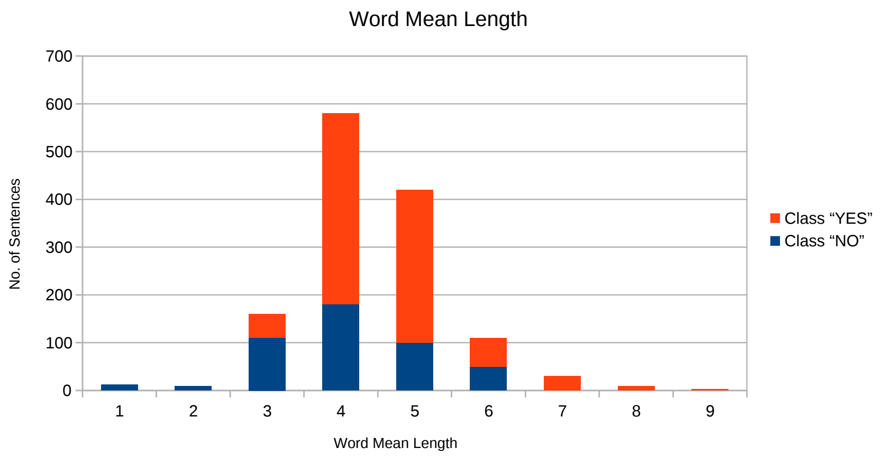


**Figure 4.7:** Argument Marker - Word Mean Length.

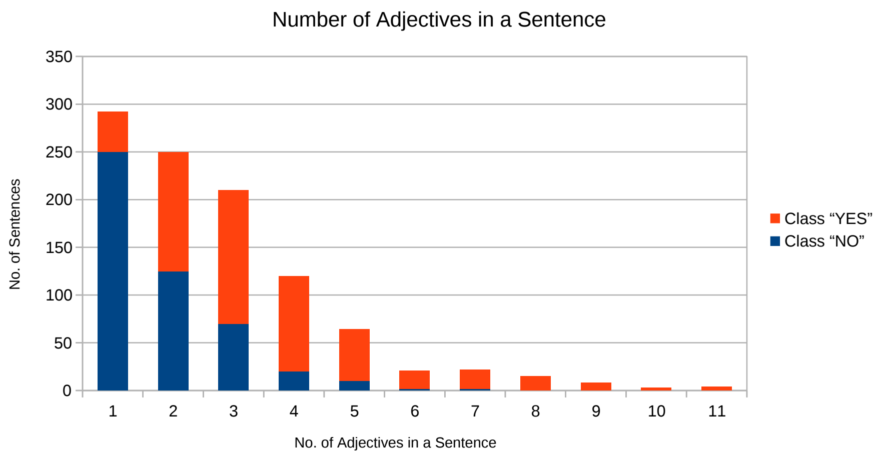


**Figure 4.8:** Argument Marker - Number of Adjectives in a Sentence.

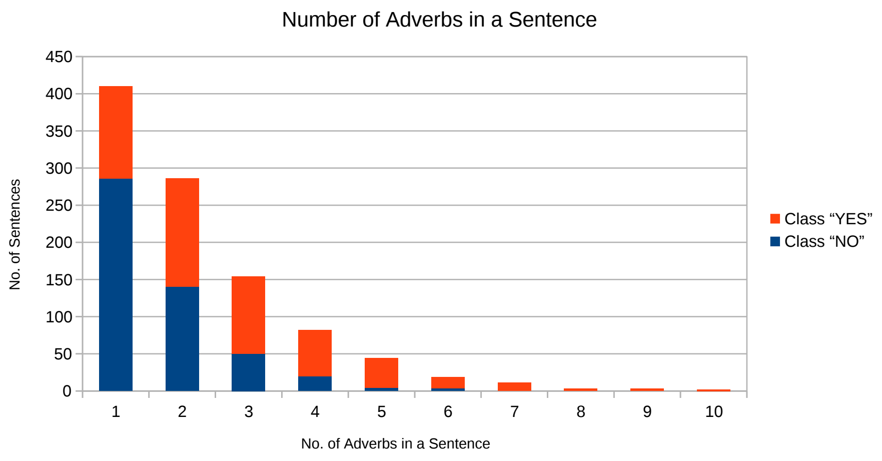


**Figure 4.9:** Argument Marker - Number of Adverbs in a Sentence.

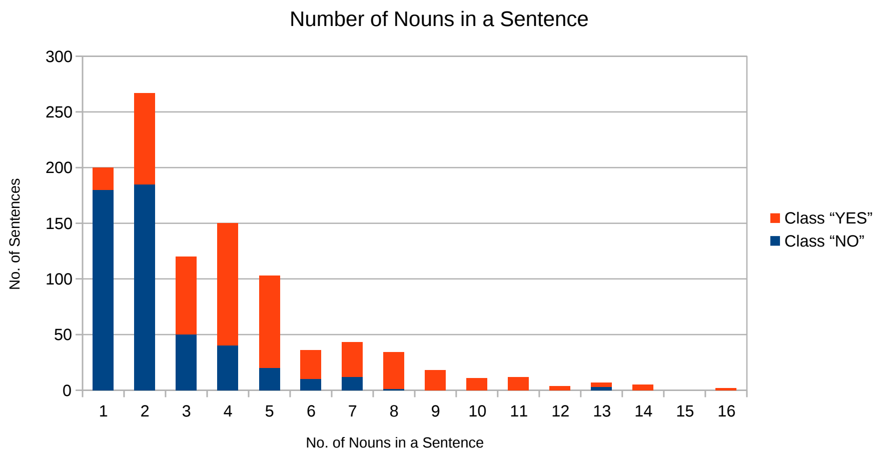


**Figure 4.10:** Argument Marker - Number of Nouns in a Sentence.

In all the above diagrams there is a common characteristic. As the frequency noted on X axis rises, we note that "Yes" class transcendent "No" class (i.e. there are more sentences in "Yes" class.

For example, in the (4.2) diagram, which shows the number of the verbs that exist in the sentences, we note that, as the number of the verbs rises, the red colour transcendents in each bar of the diagram.
But, especially for the example with the number of the verbs, there is one more interpretation, one of semantic nature. More precisely, we know that the verbs in a group of two or three sentences have a defining role in the interpretation of the meaning, as they express an action. So, the more verbs there are in a group of sentences, the more possible it is a sentence contains arguments (to be characterised as Argumentative).

But we have to note that not all the argument markers contribute in the same way in characterising a sentence as Argumentative. This is shown in the above diagrams and more precisely in those which have enough bars in which the red colour has the same size as the blue colour (there is an almost equal number of sentences between the two classes). A characteristic example is the "No. of Connecting Words in a Sentence" diagram (4.5). The markers which show this quality are used more for the fine tuning of the process.

### 4.1.2 Algorithms used in Machine Learning Procedure

In this subsection, we will see the results that the Algorithms which were used to check the accuracy of Machine Learning Process, produced. The results have been produced by carrying out classification on the Train Set and using "10 Fold Cross Validation"[1].

**Table 4.1:** Detailed Accuracy for Class "No" (Argument Extraction).

| **Algorithm** | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|
| SVM | *0.815* | *0.830* | **0.823** |
| Random Forest | **0.818** | *0.818* | *0.818* |
| Naive Bayes | *0.718* | **0.899** | *0.798* |
| Logistic Regression | *0.801* | *0.819* | *0.819* |

**Table 4.2:** Detailed Accuracy for Class "Yes" (Argument Extraction).

| **Algorithm** | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|
| SVM | *0.836* | *0.821* | **0.828** |
| Random Forest | *0.827* | **0.827** | *0.827* |
| Naive Bayes | **0.873** | *0.663* | *0.754* |
| Logistic Regression | *0.837* | *0.802* | *0.819* |

**Table 4.3:** Weighted Average on both Classes (Argument Extraction).

| **Algorithm** | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|
| SVM | **0.826** | **0.826** | **0.826** |
| Random Forest | *0.823* | *0.823* | *0.823* |
| Naive Bayes | *0.797* | *0.778* | *0.776* |
| Logistic Regression | *0.820* | *0.819* | *0.819* |

From the above tables, it is obvious that the best results were achieved by using the "Support Vector Machines" Algorithm. For this reason, we will look at some useful information about this result straight away.

[1]`https://en.wikipedia.org/wiki/Cross-validation_(statistics)`

**Table 4.4:** Additional Statistical Information (Argument Extraction).

| | **Frequency** | **Percentage** |
|---|---|---|
| Correctly Classified Instances | *838* | *82.56%* |
| Incorrectly Classified Instances | *177* | *17.4383* |
| Kappa statistic | *0.6512* | - |
| Mean absolute error | *0.1744* | - |
| Root mean squared error | *0.4176* | - |
| Relative absolute error | - | *34.90%* |
| Root relative squared error | - | *83.54%* |
| Coverage of cases (0.95 level) | - | *82.56%* |
| Mean rel. region size (0.95 level) | - | *50%* |
| Total Number of Instances | *1015* | - |

We can also note that the results which all the algorithms that we tried produced have small variations, they all produce approximately 0.8 F-Measure for both classes. A more important observation, though, is that the F-Measure for both classes ("Yes", "No") also does not have many variations. This is especially important because in the dataset we studied, we noted that there is an equal partition between the sentences that are arguments and those which are not. Also, it is important because as it results from the above tables, the algorithms have the same ability to define correctly whether a sentence belongs to class "Yes" or class "No".

### 4.1.3 Information about the Train Set

One more measurement that was performed concerns the Train Set. More precisely, in the following diagram, the "Error Rate" in the ability to correctly characterize the sentences that are being analysed, appears. We can note that by using from 30% of the Train Set (approximately 300 sentences) up to 100%, the "Error Rate" is limited between 17.5% - 18.3%. For the next diagram, "Support Vector Machines" Algorithm was used.

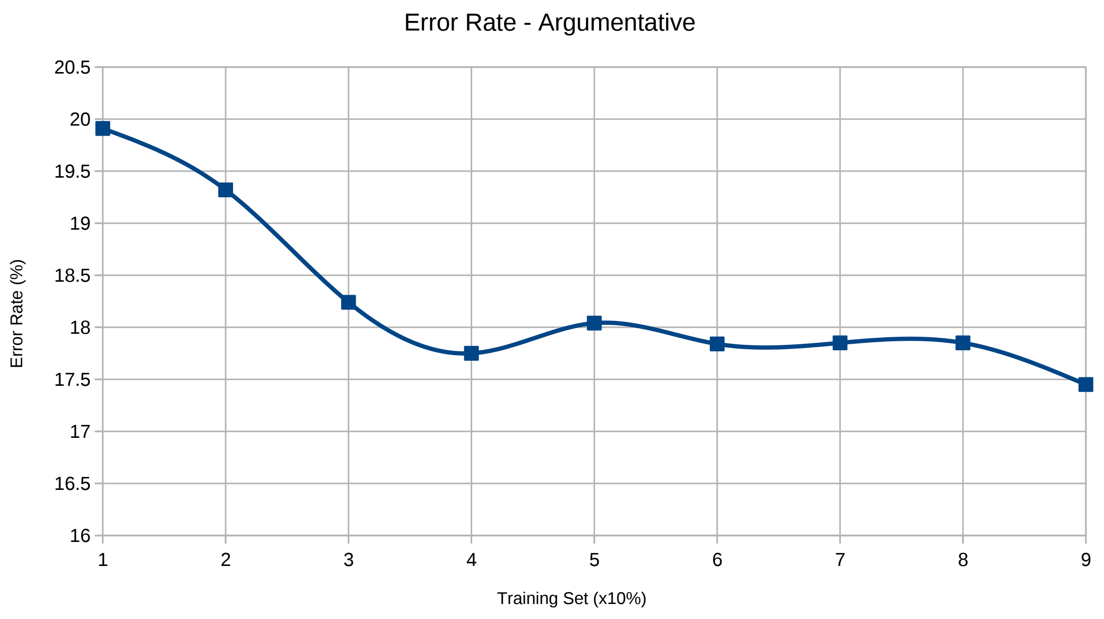


**Figure 4.11:** Error Rate of Argumentative Sentence Classification.

## 4.2 Suggestion Extraction

For the confirmation of the results for extraction of the suggestions that the writers of the texts made, we needed to create two scenarios. Initially, we carried out "10 Fold Cross Validation" (4.2.1) on the Train Set. Next, we created a second scenario, according to which we divided the Train Set so that it contains an equal number of sentences that constitute suggestions and of sentences that do not. Next, we used this set as a Train Set and performed again the Machine Learning Process (4.2.2).

### 4.2.1 "10 Fold Cross Validation" on Train Set

As described above, initially, we tried to apply "10 Fold cross validation" on the Train Set. The results are shown in the following matrix:

**Table 4.5:** Detailed Accuracy for Class "No" (Suggestion Extraction).

| **Algorithm** | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|
| J48 | *0.881* | *0.923* | *0.901* |
| Random Forest | *0.890* | *0.915* | *0.902* |
| Naive Bayes | **0.912** | *0.915* | *0.604* |
| SVM | *0.839* | **0.989** | **0.908** |

**Table 4.6:** Detailed Accuracy for Class "Yes" (Suggestion Extraction).

| **Algorithm** | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|
| J48 | *0.552* | *0.432* | *0.485* |
| Random Forest | *0.556* | *0.489* | *0.519* |
| Naive Bayes | *0.608* | **0.601** | **0.604** |
| SVM | **0.735** | *0.137* | *0.230* |

**Table 4.7:** Weighted Average on both Classes (Suggestion Extraction).

| **Algorithm** | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|
| J48 | *0.822* | *0.834* | *0.826* |
| Random Forest | *0.830* | *0.837* | *0.833* |
| Naive Bayes | **0.858** | **0.858** | **0.858** |
| SVM | *0.820* | *0.835* | *0.786* |

From the above tables, we can see that the best results were achieved by using the "Naive Bayes" Algorithm. For this reason, we will now see some useful information about this result.

**Table 4.8:** Additional Statistical Information (Suggestion Extraction).

| | **Frequency** | **Percentage** |
|---|---|---|
| Correctly Classified Instances | *871* | *85.81%* |
| Incorrectly Classified Instances | *144* | *14.19%* |
| Kappa statistic | *0.518* | - |
| Mean absolute error | *0.1901* | - |
| Root mean squared error | *0.3382* | - |
| Relative absolute error | - | *64.21%* |
| Root relative squared error | - | *87.98%* |
| Coverage of cases (0.95 level) | - | *95.67%* |
| Mean rel. region size (0.95 level) | - | *70.64%* |
| Total Number of Instances | *1015* | - |

The above results have been produced by executing "10 Fold cross validation" on the Trainset we created, whereas in the arguments section we noticed that a relatively small percentage of the sentences (almost 20%) is enough to stabilise the error rate at a low level. In this case, however, the same thing doesn't happen. We see that the algorithms that were applied have an extremely good ability to correctly characterise the sentences of the text that are not suggestions. In case, however, we want to characterise a sentence as a suggestion, there is a weakness, and low results (approximately 0.5 in the F-measure scale), are produced. This matter is especially important because it suggests that the algorithms in reality cannot characterise a sentence correctly as a suggestion or not.

The main reason why this particularity occurs (good percentages for class "No", lower percentages for class “Yes” is that the Trainset on which “10 Fold cross validation” was executed, created very few sentences that are characterised as suggestions, so only a small sample was chosen during the execution of “10 Fold cross validation”. To better understand this problem, we should think about the following.

During the execution of “10 fold cross validation”, the total of the data is divided in ten parts and then a part is chosen as a train set and the rest of the total as the test set. In our case, we had a sample of approximately 100 sentences as the train set, which contained very few sentences which are suggestions (in total there were about 170 sentences). So, in order to overcome the above problem and to be able to check if the process that was executed was correct, we tried a different tactic,which will be described in the next unit (4.2.2).

### 4.2.2 Equivalent Train Set

In the second scenario, a different Train Set was used. We divided the Train Set in a way that it contains equal amount of sentences that can be characterised as suggestions, compared to those that cannot. The reason is that the Train Set contains relatively less entries that refer to suggestions compare to the sentences that do not constitute suggestions. So, we used a corpus that contained two hundred sentences that are characterised as suggestions and two hundred more that are not. Next, we used this set as a Train Set and as a Test Set we used the whole corpus. The results are shown in the following tables:

**Table 4.9:** Detailed Accuracy for Class “No” (Suggestion Extraction, using Equivalent Train Set).

| Algorithm | Precision | Recall | F-Measure |
|---|---|---|---|
| J48 | *0.940* | *0.810* | *0.870* |
| Random Forest | **0.996** | *0.810* | **0.893** |
| Naive Bayes | *0.941* | **0.828** | *0.881* |
| SVM | *0.939* | *0.819* | *0.875* |

**Table 4.10:** Detailed Accuracy for Class “Yes” (Suggestion Extraction, using Equivalent Train Set).

| Algorithm | Precision | Recall | F-Measure |
|---|---|---|---|
| J48 | *0.470* | *0.765* | *0.582* |
| Random Forest | **0.533** | **0.984** | **0.641** |
| Naive Bayes | *0.495* | *0.765* | *0.601* |
| SVM | *0.479* | *0.760* | *0.588* |

**Table 4.11:** Weighted Average on both Classes (Suggestion Extraction, using Equivalent Train Set).

| **Algorithm** | **Precision** | **Recall** | **F-Measure** |
|---|---|---|---|
| J48 | *0.855* | *0.802* | *0.818* |
| Random Forest | **0.912** | **0.841** | **0.857** |
| Naive Bayes | *0.861* | *0.817* | *0.831* |
| SVM | *0.856* | *0.808* | *0.823* |

From the above tables we can see that the best results were achieved by using the “Random Forest” Algorithm.

**Table 4.12:** Additional Statistical Information (Suggestion Extraction, using Equivalent Train Set).

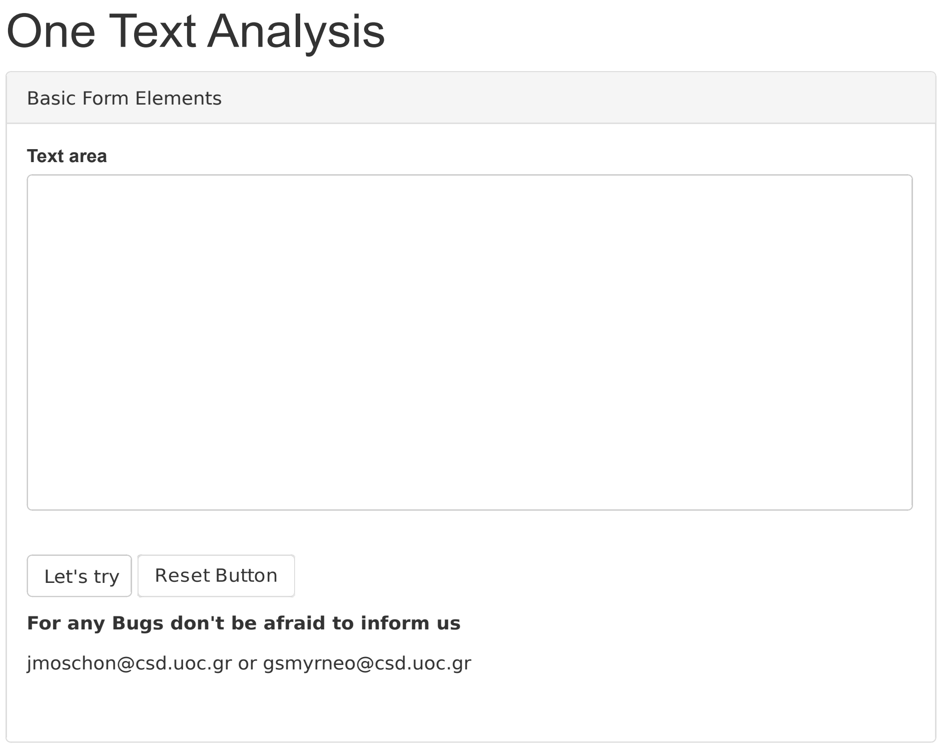
| | **Frequency** | **Percentage** |
|---|---|---|
| Correctly Classified Instances | *854* | *84.14%* |
| Incorrectly Classified Instances | *161* | *15.86%* |
| Kappa statistic | *0.5966* | - |
| Mean absolute error | *0.2273* | - |
| Root mean squared error | *0.3467* | - |
| Relative absolute error | - | *47.98%* |
| Root relative squared error | - | *73.02%* |
| Coverage of cases (0.95 level) | - | *97.14%* |
| Mean rel. region size (0.95 level) | - | *83.00%* |
| Total Number of Instances | *1015* | - |

We observe that the method we chose in order to define our train set, achieved much better results.

# 5
# Demo Application

Within the frame of this Thesis, a “Demo Application” which is divided in two distinct and independent parts, was created. The system can be tested by following the next link: `http://snf-552985.vm.okeanos.grnet.gr/opinion/`.

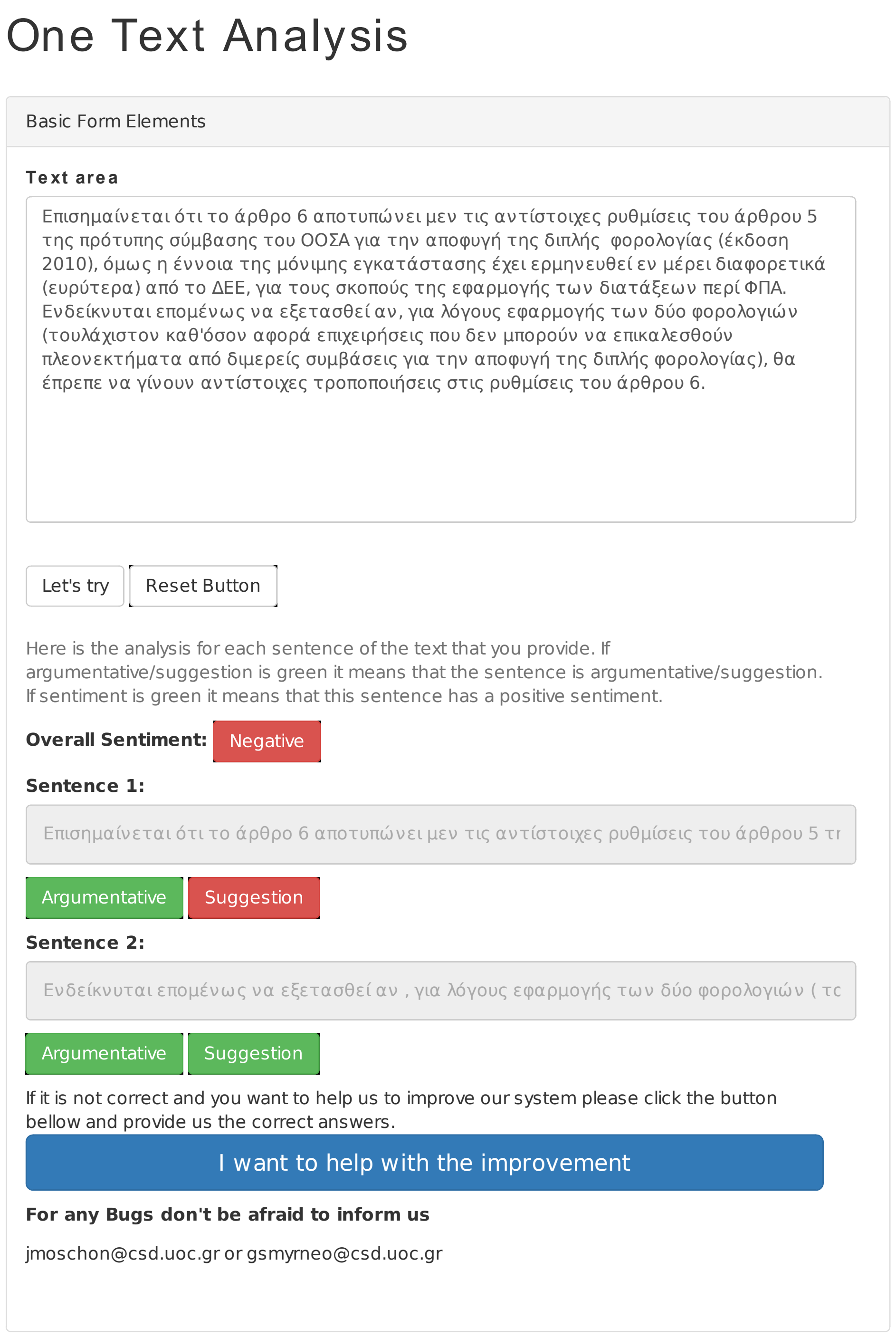
The first part is the “Back-End” which is implemented in Java SE8 following the methodology we analysed in Chapter 3, creating in that way, an HTTP server aiming at the communication of these two parts. The second part is the “Front-End”, which also constitutes the part with which the user communicates with our system, is created with the use of HTML5 and Javascript. These two parts communicate through message exchange in “JSON” form. In this demo, the user is given the opportunity to give as “entry” a text from the OpenGov Website.



For any Bugs don't be afraid to inform us

jmoschon@csd.uoc.gr or gsmyrneo@csd.uoc.gr

In the above picture we can see the initial page in which the user gives the texts that will be analysed.



One Text Analysis

Basic Form Elements

Text area

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 της πρότυπης σύμβασης του ΟΟΣΑ για την αποφυγή της διπλής φορολογίας (έκδοση 2010), όμως η έννοια της μόνιμης εγκατάστασης έχει ερμηνευθεί εν μέρει διαφορετικά (ευρύτερα) από το ΔΕΕ, για τους σκοπούς της εφαρμογής των διατάξεων περί ΦΠΑ. Ενδείκνυται επομένως να εξετασθεί αν, για λόγους εφαρμογής των δύο φορολογιών (τουλάχιστον καθ'όσον αφορά επιχειρήσεις που δεν μπορούν να επικαλεσθούν πλεονεκτήματα από διμερείς συμβάσεις για την αποφυγή της διπλής φορολογίας), θα έπρεπε να γίνουν αντίστοιχες τροποποιήσεις στις ρυθμίσεις του άρθρου 6.

Here is the analysis for each sentence of the text that you provide. If argumentative/suggestion is green it means that the sentence is argumentative/suggestion. If sentiment is green it means that this sentence has a positive sentiment.

**Overall Sentiment:** Negative

**Sentence 1:**

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 τι

**Sentence 2:**

Ενδείκνυται επομένως να εξετασθεί αν , για λόγους εφαρμογής των δύο φορολογιών ( τς

If it is not correct and you want to help us to improve our system please click the button bellow and provide us the correct answers.

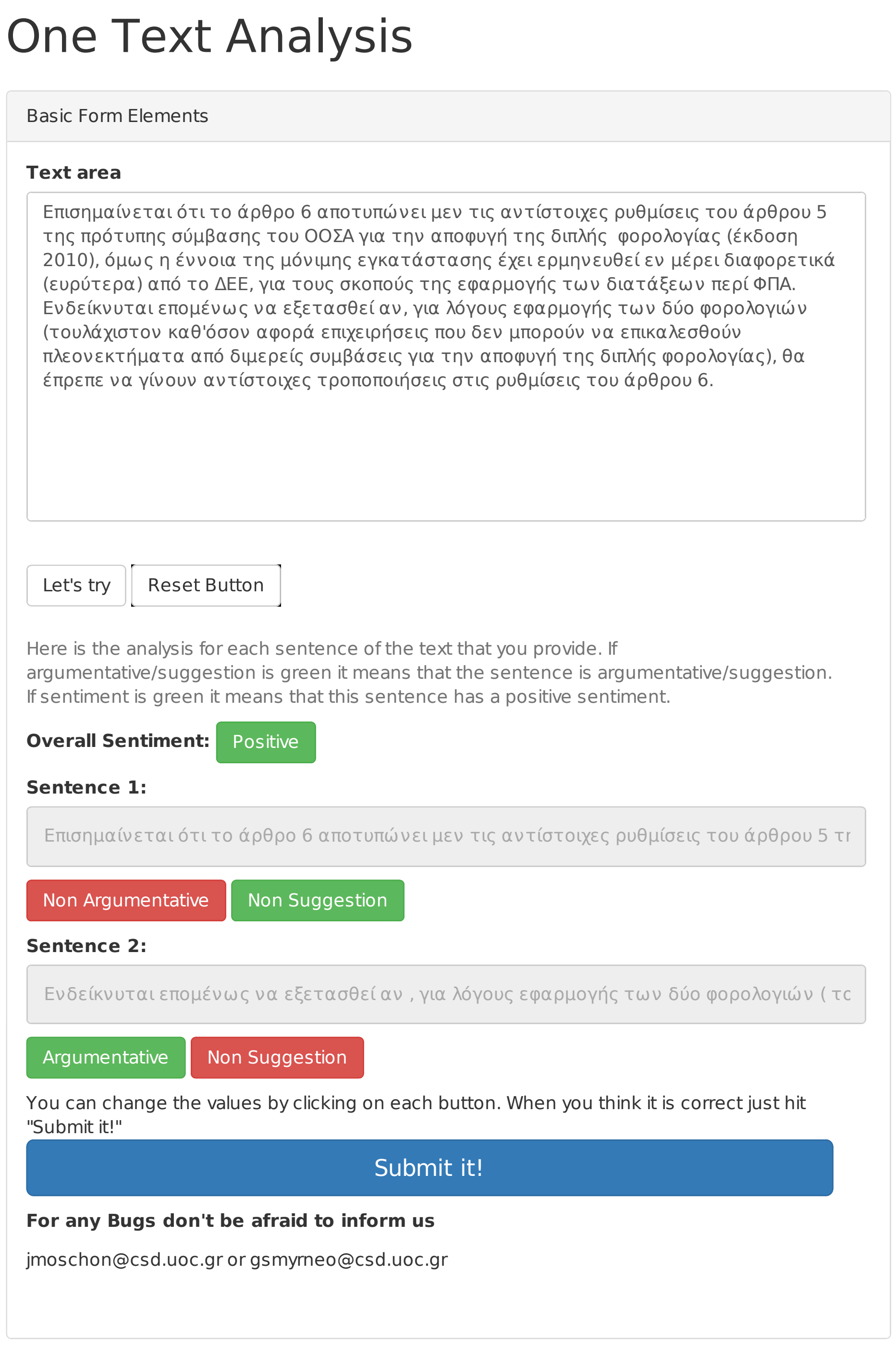
**For any Bugs don't be afraid to inform us**

jmoschon@csd.uoc.gr or gsmyrneo@csd.uoc.gr

By pressing the "Let's try" button, the results of the analysis of that specific text appear. The "Overall Sentiment" field, is red if the user is negative towards the text and green if he is positive. Then, one by one the sentences of the text appear and below each sentence an indication whether the sentence is an Argument or a Suggestion and red colour indicates that it is a non-Argument or non-Suggestion.

Another useful function that this demo application offers is that the user is being given the chance to "correct" that same System in case the results are not correct. This is done by pressing the blue button "I want to help with the improvement".

We should note that it is an important enough function, because in its total the System is based on Machine Learning.



One Text Analysis

Basic Form Elements

**Text area**

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 της πρότυπης σύμβασης του ΟΟΣΑ για την αποφυγή της διπλής φορολογίας (έκδοση 2010), όμως η έννοια της μόνιμης εγκατάστασης έχει ερμηνευθεί εν μέρει διαφορετικά (ευρύτερα) από το ΔΕΕ, για τους σκοπούς της εφαρμογής των διατάξεων περί ΦΠΑ. Ενδείκνυται επομένως να εξετασθεί αν, για λόγους εφαρμογής των δύο φορολογιών (τουλάχιστον καθ'όσον αφορά επιχειρήσεις που δεν μπορούν να επικαλεσθούν πλεονεκτήματα από διμερείς συμβάσεις για την αποφυγή της διπλής φορολογίας), θα έπρεπε να γίνουν αντίστοιχες τροποποιήσεις στις ρυθμίσεις του άρθρου 6.

Here is the analysis for each sentence of the text that you provide. If argumentative/suggestion is green it means that the sentence is argumentative/suggestion. If sentiment is green it means that this sentence has a positive sentiment.

**Overall Sentiment:** Positive

**Sentence 1:**

Επισημαίνεται ότι το άρθρο 6 αποτυπώνει μεν τις αντίστοιχες ρυθμίσεις του άρθρου 5 τη

Non Argumentative | Non Suggestion

**Sentence 2:**

Ενδείκνυται επομένως να εξετασθεί αν , για λόγους εφαρμογής των δύο φορολογιών ( τα

Argumentative | Non Suggestion

You can change the values by clicking on each button. When you think it is correct just hit "Submit it!"

**For any Bugs don't be afraid to inform us**

jmoschon@csd.uoc.gr or gsmyrneo@csd.uoc.gr

After pressing the "I want to help with the Improvement" button, as we can see in the above picture the "characterising" buttons are now in a deeper colour and the user, by pressing on them can change their colour. When he changes them and considers them to be correct, the next step is to press the "Submit it" button in order to send the corrected results back to the System.

# 6 Conclusion

In this Thesis, three self-contained mechanisms for data mining from users' texts that were published in the Greek Government's online service for Public consultation were presented. The information that is extracted by the three Mechanisms constitutes information of semantic nature. The three mechanisms for argument extraction, suggestions and overall opinion are to a great extent adapted for this particular corpus that we studied, at the same time especially for the mechanism for argument extraction, a relatively high accuracy is achieved in comparison to other studies that were taken into consideration for the completion of this Thesis. At the same, the process by which we attempted to extract suggestions to a great enough extent constitutes our own idea, given that in the past, few Systems that create that particular type of analysis had been implemented. Subsequently, up to this point, the results we received from the evaluation are very satisfying.

For the completion of this Thesis, many arguments were dealt with, which were mainly algorithmic difficulties. A lot of studying was needed to be done in the available Related Work so that we can move on to the implementation. One more big difficulty was the incomplete knowledge we had about Machine Learning. All the difficulties were dealt with by studying the available bibliography.

The first thought about Future work based on this Thesis is that further separation of Arguments could be done, as it was analysed in chapter 2. That means seeking to separate the part of the sentence in which the writer of the text claims something from the part that constitutes evaluation-validation. In the Future, we could also attempt Optimization of the suggestion extraction process, by trying other techniques. Finally, when it comes to the mechanism that extracts a writer's general knowledge on a matter, it would be important enough to test its efficiency on a better dataset.

# Bibliography

[1] Giorgos Flouris, Antonis Bikakis, Theodore Patkos, Dimitris Plexousakis. Globally Interconnecting Persuasive Arguments: The vision of the Persuasive Web. Technical Report. FORTH-ICS/TR-438, November 2013.

[2] Wandhofer, T., Van Eeckhaute, C., Taylor, S., and Fernandez, M. Wegov analysis tools to connect policy makers with citizens online. Brunel University. In Proceedings of the tGov Conference May 2012.

[3] Diakopoulos, N. A. and Shamma, D. A. . Characterizing debate performance via aggregated twitter sentiment. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems. 2010.

[4] Tumasjan, A., Sprenger, T. O., Sandner, P. G., and Welpe.Predicting elections with twitter: What 140 characters reveal about political sentiment. 2010.

[5] Theodosis Goudas Department of Digital Systems, University of Piraeus, Christos Louizos. Argument Extraction from News, Blogs, and Social Media Department of Informatics & Telecommunications University of Athens, Georgios Petasis and Vangelis Karkaletsis Software and Knowledge Engineering Laboratory.

[6] Simone Teufel. Argumentative Zoning: Information Extraction from Scientific Text. Ph.D. thesis, University of Edinburgh. 1999.

[7] Alan Sergeant. Automatic argumentation extraction. In Proceedings of the 10th European Semantic Web Conference, pages 656–660. ESWC 2013.

[8] Annotating Argument Components and Relations in Persuasive Essays C Stab,I Gurevych COLING 2014.