1.361.0

Engineering the Policy-making Life Cycle

Seventh Framework Programme – Grant Agreement 288147

Opinion Mining Prototype Evaluation, version 2

Document type:	Report
Dissemination Level:	PU
Editor:	Luis Torgo
Contributing Partners:	INESC PORTO
Contributing WPs:	WP6
Estimated P/M (if applicable):	n.a.
Date of Completion:	30 June 2014
Date of Delivery to EC	30 June 2014
Number of pages:	34

ABSTRACT

This document describes the second and final version of the evaluation of the opinion mining software component.

Authors of this document:

Luis Torgo[1], Pedro Coelho[2]

[1]: Luis Torgo
INESC Porto
email: ltorgo@dcc.fc.up.pt
[2]: Pedro Coelho
INESC Porto
email: pedro.s.coelho@inesc.pt

Contents

This page has been intentionally left blank.

1 Executive Summary

This deliverable describes the final evaluation of the Opinion Mining (OM) software prototype. This software component is part of the ePolicy decision support system and has as main goals to be able to infer the sentiment of the population concerning a set of pre-defined energy-related topics. The outcomes of this system can be explored individually by policy makers, and also be used as inputs to other components of the ePolicy decision support system.

In this deliverable we formalize the predictive tasks that are addressed by the OM prototype and present several approaches that were tried and compared within the project. We also present a series of evaluation metrics that were selected to compare the different approaches, as well as the experimental methodology that was followed to obtain reliable estimates of these metrics. The results of our evaluation and comparison of the different alternatives are presented and discussed. These results were also used to select the final models that are being applied in the OM software prototype that is running in real time at the ePolicy servers.

This deliverable is strongly related with Deliverable D 6.3 where we have described the first version of the evaluation of the OM prototype.

2 Introduction

The opinion mining (OM) component provides information on the sentiment of the population concerning a set of topics that are deemed relevant to both the global level optimizer and to the individual level simulation. However, the outcomes of the OM component can also be useful for user modelling. Namely, policy makers may find it useful to investigate the popularity trends of the topics most relevant to their job. This user-driven exploration of the sentiment of the population concerning a pre-defined set of topics is the other major goal of the OM component.

Deliverable 6.4 described the architecture (c.f. Figure 1) as well as the technical details of the OM prototype.

The overall task of the OM prototype is to be able to *infer the current sentiment of the population* concerning a *pre-defined set of energy-related topics*. To achieve this goal the OM prototype first crawls a set of *pre-defined e-participation sites* searching for new posts of the population that may discuss the selected topics. After this crawling stage the system will make *two main decisions concerning each new post*: i) which if any of the topics are discussed in the post; and ii) for each of the target topics that are mentioned, what is the sentiment expressed in the post. To make these decisions the OM prototype needs to develop (learn) models that are able to classify correctly new posts in terms of these two issues. Figure 1 shows the overall architecture of the proposed prototype. The two crucial components of the OM prototype are the **Learner** and the **Predictor**. The first is the component responsible for learning models that are able to make decisions concerning the above-mentioned two topics, whilst the second applies the learned models to new posts found in the e-participation sites.

This deliverable provides a second and final evaluation of the OM prototype focusing on the two key components mentioned above. More precisely, we report on the development and comparative evaluation of a set of alternatives for the learner and predictor components, with the goal of maximizing the overall performance of the OM prototype.

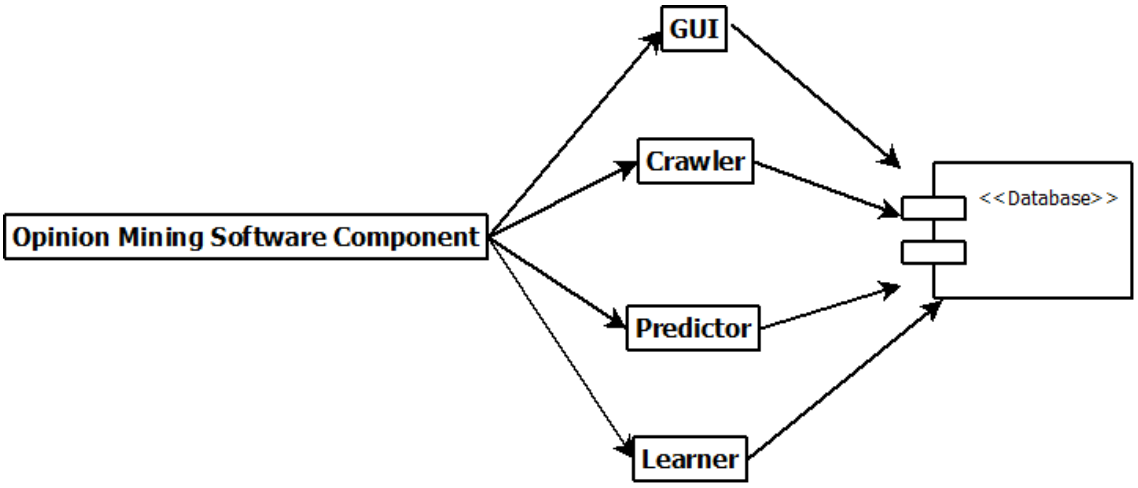

Figure 1: The proposed OM prototype architecture.

The structure of this deliverable is as follows. Section 3 formalizes the prediction tasks being addressed by the models used by the OM prototype to infer the sentiment expressed in a

text document. Section 4 describes in detail the solutions we have considered to solve these tasks. In Section 5 we describe the evaluation methodology that was selected to check the quality of the proposed solutions, while in Section 6 we present and discuss the results of this evaluation.

3 Problem Formalization

Text mining addresses the problem of analysing and extracting information from text documents. In our case the goal is to infer the sentiment expressed in each document concerning a pre-defined set of topics related to energy policies. Nowadays there is a massive amount of data available on the internet, providing an invaluable source of information on the opinion of people concerning almost every possible topic. Different e-participation tools facilitate the task of expressing our opinion. Having a system capable of classifying documents automatically will allow us to exploit massive amounts of data and extract useful information on the sentiments expressed by the public.

As we have mentioned before, given a new post, our text mining models need to be able to predict whether the post expresses sentiment concerning a set of pre-defined energy-related topics. Any of these topics may, or may not be, mentioned in the post. If a topic is mentioned then we should also infer the sentiment expressed in the post. We therefore decomposed the problem into two separate text mining tasks:

- **Topic identification** - decide for the select set of q topics which ones are mentioned in the document, i.e. make a set of q binary decisions.
- **Sentiment scoring** - for each topic that is mentioned in a document decide what is the sentiment concerning that topic on a pre-defined scale.

The two tasks mentioned above can be seen as instances of predictive tasks. Predictive modelling has the general goal of inferring the value of some set of unknown variable(s) $\mathbf{y}$ from the values of known variables $\mathbf{x}$ that are supposed to influence the values of the $\mathbf{y}$. Predictive tasks can be described as data analysis problems where one assumes that there is a functional dependency between a target variable Y and a set of descriptor variables (or predictors) $X_1, X_2, \cdots, X_p$. The goal of predictive modelling is to infer this function from a sample of mappings between values of the predictors and the target variable, i.e. a (*training*) data set $\{\langle \mathbf{x}_i, Y_i \rangle\}_{i=1}^{N}$, where $\mathbf{x}$ is a *feature vector* formed by values of the p predictor variables $X_1, X_2, \cdots, X_p$. In our concrete application the information available for the models to make their decisions is in the form of a text document (a post at some web site). This means that if we transform the information in a text document to a feature vector representation, we can look at our tasks as instances of standard predictive tasks.

In data mining the two most common instances of predictive tasks are known as regression and classification. In regression we use the provided training data set to induce a model of the unknown function,

$$Y = f(\mathbf{x}) \tag{1}$$

where Y is the **numeric** target variable and $\mathbf{x}$ is the vector of predictor variables $X_1, X_2, \cdots, X_p$.

In classification we have a similar inference problem but the domain of the target variable is a finite set of labels, i.e. Y is a **nominal** variable.

Our first task of topic identification is an instance of a learning problem known as multi-label classification (e.g. [10]). Namely, for each of the q considered topics we need to decide if the topics are or not mentioned in a post, i.e. q binary decisions. Multi-label classification can be addressed in several ways [10]. Among the most frequent alternatives we have the so-called *binary relevance* approaches that divide the task into q standard binary classification tasks. This is one of the approaches that we will take to the topic identification problem within e-Policy. The main advantage of these approaches is the fact that they allow the use of existing classification methods by simply pre-processing the available data to have q different training sets, one for each of the q topics. More complex approaches involve the modification or creation of special-purpose algorithms that handle multi-label classification tasks directly. We will also try this type of approaches. The main disadvantage of these approaches is the fact that there are not so many algorithms able to handle these tasks. Irrespectively of the approach, the task in multi-label classification consists of obtaining a model of the following function,

$$\mathbf{y} = f(\mathbf{x}) \tag{2}$$

where $\mathbf{y}$ is a vector of target variables $Y_1, Y_2, \cdots, Y_q$; and $\mathbf{x}$ is the vector of predictor variables $X_1, X_2, \cdots, X_p$.

We address the task of sentiment scoring as q separate score prediction tasks. In theory we could also consider the hypothesis of having as output of this sentiment scoring function as a vector of q scores. However, we have decided not to proceed that way given that we do not think there is a correlation between the sentiment scores of each topic, and thus there is not particular advantage of using these alternatives. The sentiment on a certain topic can be expressed in many ways. Usual formats include positive vs negative sentiment, or some rating scale. We follow the latter approach by trying to infer the sentiment in a document in terms of a $-2, -1, 0, 1, 2$ scale, where negative numbers represent negative sentiment, while positive numbers the opposite. Coarser or finer granularities would be possible, but the approaches we will describe are generalizable to these other solutions as long as they can be regarded as values of an ordinal variable.

Given that our target variable is the value of the sentiment on an ordered fixed scale, i.e. an ordinal scale, we have a particular type of prediction task that differs from the more standard regression and classification tasks. Still, given the limited amount of available methods that address ordinal target variables, we have solved this sentiment scoring task using regression and classification approaches. In order to do so, we have designed pre-processing steps that will be described in the next section.

4 Implemented Solutions

In the previous section we have identified and formalized two main prediction tasks as driving the key components of the OM prototype: i) topic identification; and ii) sentiment scoring.

Both tasks share the same input - a set of documents. In more formal terms this means that all tasks share the same predictor variables, the only difference being on what they predict

(i.e., the target variables of the prediction models).

In this section we describe how we address the two tasks. As a first step, we present the data representation that translates the text document into a set of variable values (a feature vector), required by both tasks.

4.1 Document Representation

The way we represent a document can have an impact on the obtained models and on the respective predictive performance. The literature describes several ways of representing a text document as a feature vector, with the most popular alternatives being the Bag of Words (BOW) and the N-gram representations.

The N-gram representation involves the creation of sequences of N-words. For example, on a 2-gram representation, the sentence 'I went to the garden today' could generate 3 groups of 2-grams, 'I went', 'to the', 'garden today'. Then, after discovering all the groups in our corpus, we count how many times they appear in the document and assign this value to the group. This type of representation tries to keep some information about the sequence of the words or the context in which each word appears.

The Bag of Words (BOW) representation, the one we have adopted, is the most frequent approach. We represent the document by separating the sentences into single words. For example, on the previous referred sentence, we can identify the words 'I', 'went', 'to', 'the', 'garden' and 'today'. This strategy usually proceeds by identifying all words in a given corpus (eventually after some pre-processing steps like stop word removal, or word stemming) and then by counting the occurrences of each identified word on each document. This means that the features or predictor variables used to represent the texts in a data set will be this (often large) set of identified words. As values of these predictors an usual choice is to assign the frequency (the number of times the word appears on the document, or term frequency (tf)). Another option is the tf-idf (term-frequency inverse-document-frequency) score which modifies the term frequency with a factor related to the importance of each word (term) of a document within a collection of documents. If the word appears more frequently in the collection of documents then its tf-idf value will be high. This tells us which words separate documents better (if they only appear in few documents then they distinguish these from the others).

On both representations, we need to decide what to do with all the words found in a corpora. Do all of them interest us? Should we, for example, keep numbers and punctuation? Although some of these decisions may be domain-dependent, frequent pre-processing stages include: (i) removal of stop words; (ii) removal of punctuation and numbers; and (iii) word stemming.

In summary, although many alternatives exist for representing the information in a text document we have selected the frequently used bag of words representation using term frequency as values. We have also opted to remove stop words, punctuation and numbers and apply word stemming. In order to reduce the number of words, we have removed sparse terms with a factor of less than 0.95. This resulted in using a total of 172 words whose frequency

will be the predictor values describing each text document.

4.2 Topic Identification

The first step of the OM prototype for inferring the sentiment expressed in a post consists in identifying the topics that are addressed in this text document. In Section 3 we have mentioned that we have tried two approaches to this identification task.

The first approach consists of handling this as q separate binary classification problems, where q is the number of pre-selected topics. For each of these q prediction problems a binary classification model was developed using the available training data with the goal of approximating the function $Y = f(\mathbf{x})$, where Y takes two possible values, e.g. yes and no, meaning that the respective topic is (or is not) mentioned in the document being analysed that is represented by the feature vector $\mathbf{x}$ (as we have seen the frequency of 172 pre-selected words). To address these q binary classification tasks we have learned several alternative models using different machine learning algorithms that will be detailed in Section 5.2.

The second approach tackles the q topics identification problem using a single multi-label classification model. The idea/motivation is to try to explore eventual correlations among the q topics. With this purpose we tried different variants of the Clus [1] system.

4.3 Sentiment Scoring

The second step of the OM prototype is to infer the sentiment expressed in each document that was identified as mentioning a certain topic.

The selected scoring scale can be regarded as the domain of an ordinal variable. As mentioned before, few algorithms are available to address predictive tasks with ordinal target variables. In this context, we have implemented a different approach not to limit the range of solutions to this task. Namely, we have followed two different paths to the q sentiment score prediction tasks: i) predict the score using classification models; and ii) using regression models.

Classification algorithms do not assume any ordering of the values of the target variable, which we have seen is not true in our sentiment scale. An order among the values means that it is worse to misclassify a document with sentiment -2 as having sentiment 2, than classifying it as having sentiment -1. Classification algorithms consider all errors equally serious and thus can not cope with the above distinction. To achieve this distinction we can resort to cost matrices. A cost matrix is a $c \times c$ matrix where c is the number of possible labels of the target variable. The rows and columns of this matrix represent the possible values for the predictions and true values of any test case. The entries in the matrix specify a value (a cost) for each possible combination of predicted and true target variable value. Using these matrices we can specify the costs such that it is more costly for the model to predict a value of 2 for a document with true sentiment of -2, than the cost of predicting -1. This means that through cost matrices we can convey the order information to the classification models by means of different costs of the errors, and thus use the large variety of classification algorithms in our q sentiment scoring tasks.

Regression tasks assume that the target variable is numeric, which means that there is an implicit ordering among its values. This allows us to handle the different types of sentiment scoring errors naturally without having to resort to cost matrices as in classification. Still, regression methods allow interpolation among values, which means that some model could come up with a predicted sentiment score of 1.234. In order to force the predictions into our selected sentiment scale, when using regression tools, we will re-scale the predicted values back to the original scale by applying the following rounding function to the predictions:

$$r(x) = \begin{cases} 2 & \text{if } x \geq 1.5 \\ 1 & x \in]0.5, 1.5] \\ 0 & x \in]-0.5, 0.5] \\ -1 & x \in]-1.5, -0.5] \\ -2 & \text{if } x < -1.5 \end{cases} \tag{3}$$

In summary, we have tried, evaluated and compared different variants of several classification and regression algorithms (c.f. Section 5.2) with the goal of solving sentiment scoring for each of the q topics. When one of the q topics is identified as being present in one new post using the models of the previous section, the respective sentiment scoring model is used to forecast the sentiment score expressed in that document regards that topic.

5 Evaluation Methodology

This section describes the key aspects of the methodology that was followed to evaluate the proposed solutions. We start by describing the data that was available for this evaluation. We then provide more details concerning the models that were used in implementing the solutions outlined in Section 4. Finally, we discuss the metrics that were used to evaluate the different alternatives as well as the experimental methodology that was followed to obtain reliable performance results.

5.1 The e-Policy data set

The e-Policy project is concerned with energy policies for the region of Emilia-Romagna in Italy. In this context, all activities concerning the involvement of the population with e-participation tools will naturally use the Italian language. Most of the existing research on text mining is carried out with the English language but work on other languages is growing [2]. Especially in huge global events such as the Olympics or Soccer World Championships, it is very important for the media to be able to extract and process large amounts of data as fast as possible which makes the study and development of this field very important in all languages. On the e-Policy project, having efficient models and tools tailored for the Italian language is essential.

In terms of the goals of opinion mining within the project the consortium has decided to focus on 14 main topics and 3 subcategories (economic, environmental and technological aspects) for each, totalling 42 topics. The goal of the tools developed within the project is to infer the sentiment of the population concerning these 42 topics and also to provide information on

tendencies of this sentiment along time, so that the eventual impact of decisions taken by policy makers can be measured. The list of 14 selected main topics is the following:

- Photovoltaic
- Thermal
- Wind power
- Hydroelectric
- Biomass
- Geothermal
- Biogas
- Fusion
- Biofuels
- Eco-Mobility
- Combustion
- Free energy
- Energy saving
- Waste to energy

As mentioned above, for each of these 14 topics, 3 different aspects were considered.

In terms of sources for mining the opinion of the public, the consortium has decided to use as testbeds two Italian websites [7, 8] - Energetic Ambient (Figures 2 and 3) and the Newclear blog (Figure 4). On both websites the different posts are structured as a hierarchy starting with a top post and then sub-sequent posts discussing this main post.

Figure 2: Energetic Ambient front page [7].

Figure 3: Energetic Ambient forum.

In the context of the OM prototype implementation we have developed crawlers for these two websites that have collected a data set with posts and some information associated with each post. Table 1 presents the information that is collected for each post by our crawlers, like the date, title and post counter of each post (if it is a main post or a reply to the main post), etc. In spite of the availability of all this information, the approaches described in this deliverable will only make use of the text of each post.

Predictive modelling requires a training set where the values of the target variables are known. In the context, we need a data set with posts that are tagged regards the sentiment expressed for each of the topics selected for this study. Tagging a large amount of posts for the 42 topics is a task that requires huge human resources with expertise in the energy field. Carrying out this task for the roughly 600 000 posts[1] that currently form our data base is not possible with the human resources available to the project. In this context, the amount of data (posts) that were read, analysed and tagged concerning the sentiment is limited, and amounts currently to around 800 text documents (c.f. Table 1). Moreover, for the same reasons related with limited human resources, we had to limit our experiments to 3 out of the 42 topics : 'Photovoltaic economic aspects', 'Photovoltaic environmental aspects' and 'Photovoltaic technology aspects'. Table 2 indicates the number of available posts for each of the selected topics.

In summary, for the initial task of topic identification we will have a data set of formed by 857 posts tagged for the 3 topics. For the subsequent sentiment scoring tasks we will have data sets with different sizes depending on the topic, according to the numbers on Table 2. These numbers are obviously small, which may have an impact on the predictive accuracy of the obtained models. Still, if more tagged documents become available it is easy to re-learn new models that will potentially improve the predictive accuracy of the OM prototype. Nevertheless, we should mention that the feedback we have collected from human experts concerning the predictions of the models regards concrete posts, is generally very positive.

[1]This number is growing constantly as our crawlers are running in real time.

Figure 4: Newclear blog [8].

Table 1 e-Policy Data set composition.

Number of Documents	Number of Tagged Documents	Features
582382	857	ID,Author ID, Title, Text, Date, Postcounter, URL, Blogname, Topic, Score

Table 2 e-Policy data set composition by topic.

Topic	Number of Tagged Documents
Photovoltaic Economic Aspect	501
Photovoltaic Environmental Aspect	63
Photovoltaic Technology Aspect	410

5.2 The Modelling Techniques

The predictive tasks we have described in Section 4 involve three types of models: i) multi-label classification; ii) classification; and iii) regression.

Concerning multi-label classification we have considered in our experiments the Clus [3] system. This learning algorithm is a decision tree and rule induction system that implements the predictive clustering framework described in [3]. While most decision tree learners induce classification or regression trees, Clus generalizes this approach by learning trees that are interpreted as cluster hierarchies. Such trees are called predictive clustering trees or PCTs. This system can handle multi-label classification tasks. We have considered several variants of this tool in our evaluation. Namely, we tested different ensemble parameters with the options Bagging, Random Forests, Random Subspaces and Bag of Subspaces. As for the other parameters of Clus, they were left with the default settings.

Concerning standard classification and regression tasks we have considered some of the most popular techniques: Random Forests, Support Vector Machines and Neural Networks. These

approaches not only are recognised as some of the best modelling tools as are able to address both classification and regression tasks.

poRandom Forests [4] are an ensemble learning method for classification and regression tasks composed of many decision trees created with the training data. Each tree is trained on a bootstrapped sample of the original dataset and each time a split node is created, only a randomly chosen subset of the predictors are considered for splitting. In terms of using random forests for prediction, their forecasts are the mode of the classes output by each tree in the ensemble in the case of classification tasks, or the average of the predicted values if it is a regression problem. In our experiments we have used the implementation available in the R package 'randomForest', ported from the original Fortran code by Andy Liaw and Matthew Wiener [9]. In terms of variants of these models we have considered different values for the parameter *ntree* which controls the number of trees to grow, and the parameter *mtry* that controls the number of variables randomly sampled as candidates for each split.

Support Vector Machines [5, 6], or SVMs, are a relatively recent modelling approach with a large success in many application domains. This approach is applicable to both classification and regression tasks. Nevertheless, the approach was originally developed for binary classification problems and it is easier to explain it within this setup. SVMs try to find a hyperplane that separates the cases belonging to each class (as for instance linear discriminants also do). With the goal of finding the hyperplane that maximizes the margin between the cases of the two classes, SVMs use quadratic optimization algorithms. Unfortunately, most real world problems are not linearly separable. The solution provided by SVMs to this problem consists in mapping the original data into a higher dimension input space where the cases belonging to the two classes can already be linearly separable. Although this solves the problem of linear separability, this creates another problem - applying the quadratic optimization algorithms on these high dimension spaces is computationally very demanding. To solve this extra problem SVMs use what is known as the kernel trick, which consists in using certain kernel functions that are cheap to compute and that are proven to lead to the same result as the expensive dot products that are used in the quadratic optimization algorithms when applying them in the high dimension space. These kernel functions are cheap to compute because they are calculated in the original, low dimension space. Still, their result is equal to the mentioned dot products which allows SVMs to obtain the hyper-planes in the high dimension space without having to carry out heavy computation steps on this space. This general approach has been generalized to both multi-class problems and regression tasks, and thus we can use this methodology in our tasks. We have used the SVM implementation available in the R package 'e1071' created by David Meyer [12]. In terms of different variants of SVMs we have we have varied the parameters *cost*, *epsilon* and *gamma*. The parameter *cost* sets the value associated with the cost of constraints violation, it is the 'C'-constant of the regularization term in the Lagrange formulation. The parameter *epsilon* controls the epsilon in the insensitive-loss function and *gamma* is a parameter used in the kernel.

Artificial Neural Networks [11, 15] are models with a strong biological inspiration. They are composed by a set of units (neurons) that are connected. These connections have an associated weight and the learning process consists of updating these weights. Each unit has an activation level and means to update this level. Some of these units are connected to the out-

side, being called input and output neurons. Each unit has one simple task, receive the input impulses and calculate its output as a function of these impulses. This calculation is divided in two parts: a linear computation of the inputs and a non-linear computation (activation function). Different activation functions provide different behaviours. Some examples of common functions are the Step function, the Sign function and the Sigmoid function. The units can also have thresholds that represent the minimum value of the weighted sum of the inputs that activates the neuron. There are two main types of Artificial Neural Networks:- the feed-forward networks and the recurrent networks. The feed-forward networks have uni-directional connections (from input to output), without cycles, while the recurrent networks have arbitrary connections. Usually the networks are structured in layers. On a feed-forward network each unit is connected only to units on the following layers while on a recurrent network this does not happen and the network can have feedback effects, possibly exhibiting chaotic behaviour. They usually take longer to converge. The learning process of Artificial Neural Networks consists of updating the weights of the connections. The most popular way to do this is by using the Backpropagation algorithm. Each example is presented to the network. Then, if the output produced is correct, nothing is done. If it is not correct then we need to re-adjust the network weights. In networks with multiple layers the adjustment is not simple as we need to divide the adjustments across the nodes and layers of the network. A detailed description of the back-propagating algorithm is given by David E. Rumelhart [13]. In our experiments we have used the implementation of feed-forward Artificial Neural Networks available in the R package 'nnet' created by Brian Ripley [14]. In terms of different variants of ANNs we varied the parameter *size* that controls the number of units in the hidden layer, and the parameter *decay* which controls the weight decay.

5.3 Evaluation Metrics

Our evaluation has the goal of comparing different approaches to 2 tasks: i) multi-topic identification; and ii) sentiment scoring. We have seen that these two tasks are inherently different. This is also reflected on the choices we have made concerning the metrics used to characterize and compare the performance of the different alternatives.

For the multi-topic identification task the goal is to predict for the three selected topics if a document mentions them or not. This means that for each document we want to compare the prediction ($\hat{\mathbf{y}}$) of our approach (a vector of three binary decisions concerning the three topics, e.g. $\langle yes, no, yes \rangle$) against the ground truth ($\mathbf{y}$), another vector of three decisions.

There are several metrics (e.g. [10]) that can be used to evaluate the performance of a model (whatever the followed approach) on a multi-label classification task like ours. In our experiments we have selected a sample of the most popular metrics. The following equation describes the *Hamming Loss*,

$$HammingLoss = \sum_{i=1}^{N_{test}} \frac{1}{q} |\hat{\mathbf{Y}}_i \, \Delta \, \mathbf{Y}_i| \tag{4}$$

where, N_{test} is the number of test cases, q the number of topics, $\hat{\mathbf{Y}}_i$ is the set of topics that are

predicted as present by the model, $\mathbf{Y}_i$ is the set of topics that are present in the document, and Δ is the symmetric difference of two sets[2].

For instance, if the prediction of a model for a document concerning topics t_1, t_2 and t_3 is $\langle yes, no, yes \rangle$, then $\hat{\mathbf{Y}}_i = \{t_1, t_3\}$. Moreover, if the true value for the document is $\langle no, no, yes \rangle$ then $\mathbf{Y}_i = \{t3\}$, which means that $\hat{\mathbf{Y}}_i \, \Delta \, \mathbf{Y}_i = \{t_1\}$.

Another popular measure is the multi-label *Accuracy* defined as,

$$Acc = \sum_{i=1}^{N_{test}} \frac{|\hat{\mathbf{Y}}_i \cap \mathbf{Y}_i|}{|\hat{\mathbf{Y}}_i \cup \mathbf{Y}_i|} \tag{5}$$

Precision in the context of multi-label classification is defined as,

$$Prec = \sum_{i=1}^{N_{test}} \frac{|\hat{\mathbf{Y}}_i \cap \mathbf{Y}_i|}{|\mathbf{Y}_i|} \tag{6}$$

whilst *Recall* is defined as,

$$Rec = \sum_{i=1}^{N_{test}} \frac{|\hat{\mathbf{Y}}_i \cap \mathbf{Y}_i|}{|\hat{\mathbf{Y}}_i|} \tag{7}$$

With these two latter metrics it is common to aggregate their values into the so-called *F1 score*,

$$F_1 = \sum_{i=1}^{N_{test}} \frac{2 \times |\hat{\mathbf{Y}}_i \cap \mathbf{Y}_i|}{|\hat{\mathbf{Y}}_i| + |\mathbf{Y}_i|} \tag{8}$$

Regarding the sentiment scoring task we have to take into account that our target is an ordinal variable. Moreover, as we have seen we will consider both regression and classification algorithms to solve this task. Still, independently of the algorithm their predictions can be cast into the selected scale of sentiment. For comparing the true and predicted sentiment score of a document we have used a cost matrix that can express the notion that not all errors are equivalent. We have used as evaluation metric the total cost of the predictions. This evaluation metric assumes the existence of a cost matrix indicating the cost of each misclassification. Models should try to minimize this score. We have used the following cost matrix in our experimental comparisons:

[2]The symmetric difference of two sets is the set of elements which are in either of the sets and not in their intersection.

Table 3 Cost matrix used in our experiments.

	-2	-1	0	1	2
-2	0	1	2	3	4
-1	1	0	1	2	3
0	2	1	0	1	2
1	3	2	1	0	1
2	4	3	2	1	0

Given this cost matrix the total cost of the predictions of a model for a single topic, given a test set with N_{test} documents is given by,

$$TC = \sum_{i=1}^{N_{test}} M_{\hat{y}_i, y_i} \tag{9}$$

where $M_{\hat{y}_i, y_i}$ is the entry in the cost matrix M corresponding to a prediction of $\hat{y}_i$ for the document whose true value is y_i.

5.4 Experimental Methodology

Any evaluation procedure based on data must be concerned with the statistical significance of the reported results. With this goal in mind we have designed an experimental methodology that can provide reliable estimates of the evaluation metrics that were described in the previous section, and that will also allow for comparisons of the observed performance differences in terms of statistical significance levels.

The data sets to be used in our experimental comparison are different depending on the task being addressed. Still, for each of the problems, all considered model variants will be evaluated using the same train and test partitions of the available data. More specifically, for each predictive task we will estimate the performance of all alternatives included in our study by means of 10 repetitions of a 10-fold Cross Validation process. This means that all scores we will report are averages of 100 train+test trials with the respective modeling solution. This experimental procedure ensures a good level of statistical significance of our reported results.

6 Results of the Evaluation

In this section we present and analyse the results of our experimental evaluation of the developed models. We will present the results separately for each of the tasks: (i) topic identification; and (ii) sentiment scoring.

6.1 Topic Identification Results

The different techniques that were described in Section 5.2 were compared in the task of identifying the topics that are present on a set of documents using the evaluation metrics that we

have presented in Section 5.3. These comparisons were carried out using the experimental methodology detailed in Section 5.4.

Given the amount of model variants that were considered (c.f. Annex A) we are going to present the overall results of a subset of these variants. Namely, we have selected one representative of each of the modeling algorithms that we have used - random forests, support vector machines, neural networks and the Clus approach. For each of these algorithms we will analyze the performance of the variants that achieved the best average F_1 score in our experiments. Table 4 shows the results of these best variants in terms of each of the evaluation metrics. For each model we present the average score of each metric across the 100 repetitions (10 $\times$ 10-fold cross validation) as well as the respective standard deviation.

Overall, the results of Table 4 indicate that the best performance was achieved by a variant of a random forest, closely followed by the best variant of SVMs[3]. These results also provide evidence that for this particular application there is no advantage in using a special-purpose multi-label classification algorithm like Clus. In effect, the results of this approach are generally worse than the results obtained with strategies that treat each topic identification separately.

Figures 5 to 9 provide a more detailed analysis of the performance of these best variants by showing the distribution of the scores on the 100 repetitions through box plots. Once again we confirm the good and stable performance of the random forest in this topic identification task.

Summarizing, the performance achieved by the models we have considered is generally very satisfactory. Having a precision around 75% indicates that in general our models are very accurate when they say that some topic is referred in a document, i.e. there are few false positives. Moreover, with a recall around 70% we conclude that these models do not miss many documents, i.e. there are few false negatives. This good performance was also subjectively confirmed by human experts on the subject that have analyzed some of the documents classified by our models and have considered that the performance was generally very good and accurate.

[3]Please note that all metrics except $HammingLoss$ are to be maximized.

Table 4 Best performing models.

Measure	Learner	Average	Standard Deviation
Hamming	randomForest.v8	**0.22**	0.03
	svm.v6	**0.22**	0.03
	nnet.v4	0.24	0.03
	clus.v1	0.26	0.03
Accuracy	randomForest.v8	**0.68**	0.04
	svm.v6	0.66	0.05
	nnet.v4	0.62	0.04
	clus.v1	0.62	0.04
Precision	randomForest.v8	**0.76**	0.04
	svm.v6	0.70	0.05
	nnet.v4	0.70	0.06
	clus.v1	0.63	0.04
Recall	randomForest.v8	**0.73**	0.04
	svm.v6	0.71	0.05
	nnet.v4	0.66	0.04
	clus.v1	0.68	0.05
F1	randomForest.v8	**0.72**	0.04
	svm.v6	0.69	0.05
	nnet.v4	0.66	0.04
	clus.v1	0.64	0.05

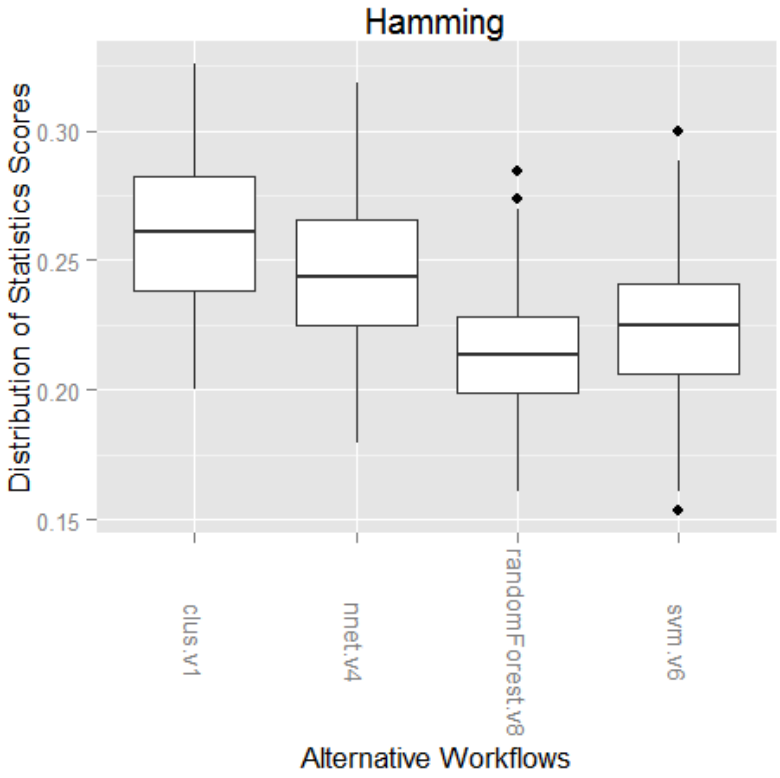

Figure 5: Hamming measure of the models in the experiments. .

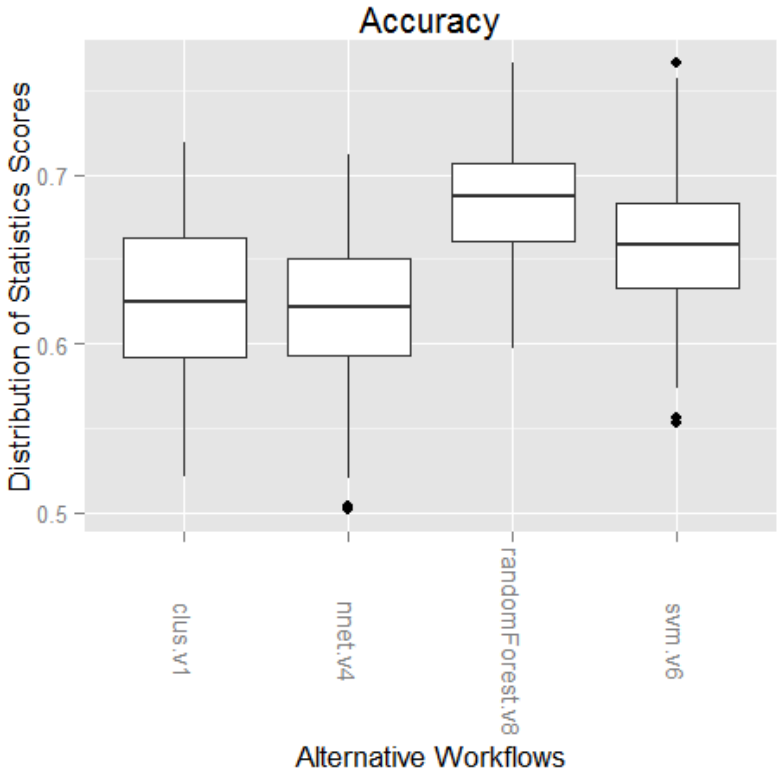

Figure 6: Accuracy measure of the models in the experiments.

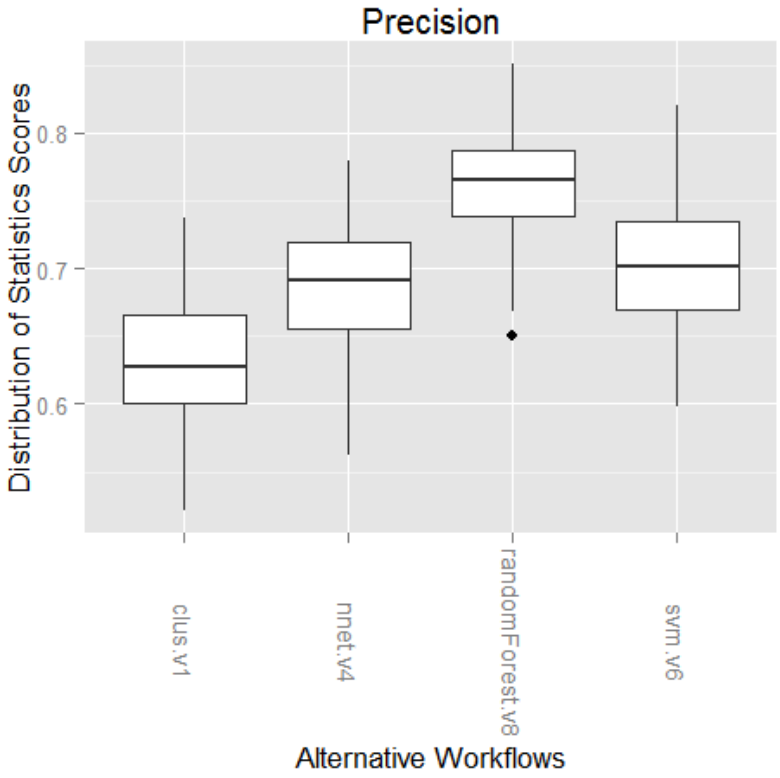

Figure 7: Precision measure of the models in the experiments.

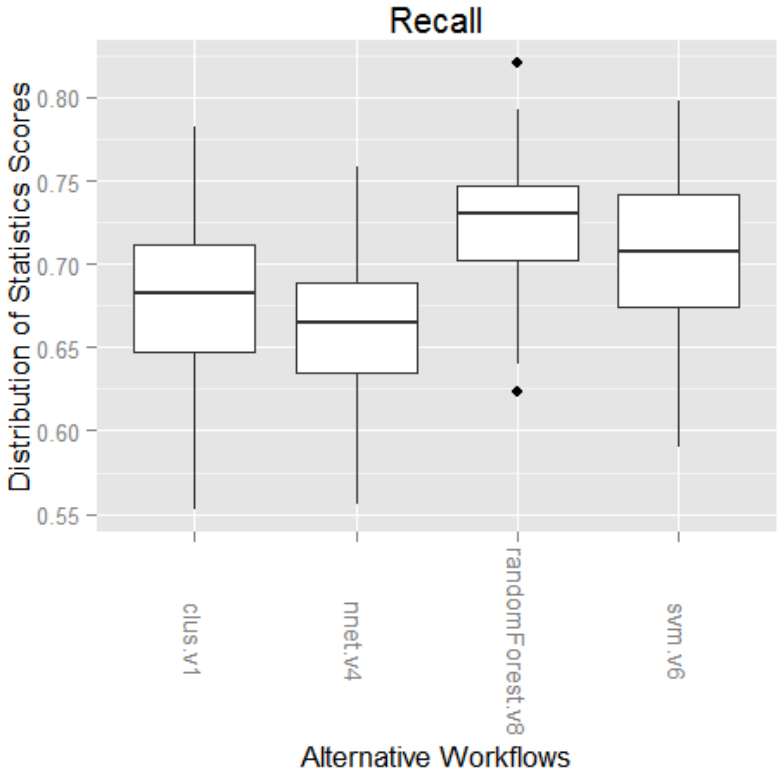

Figure 8: Recall measure of the models in the experiments.

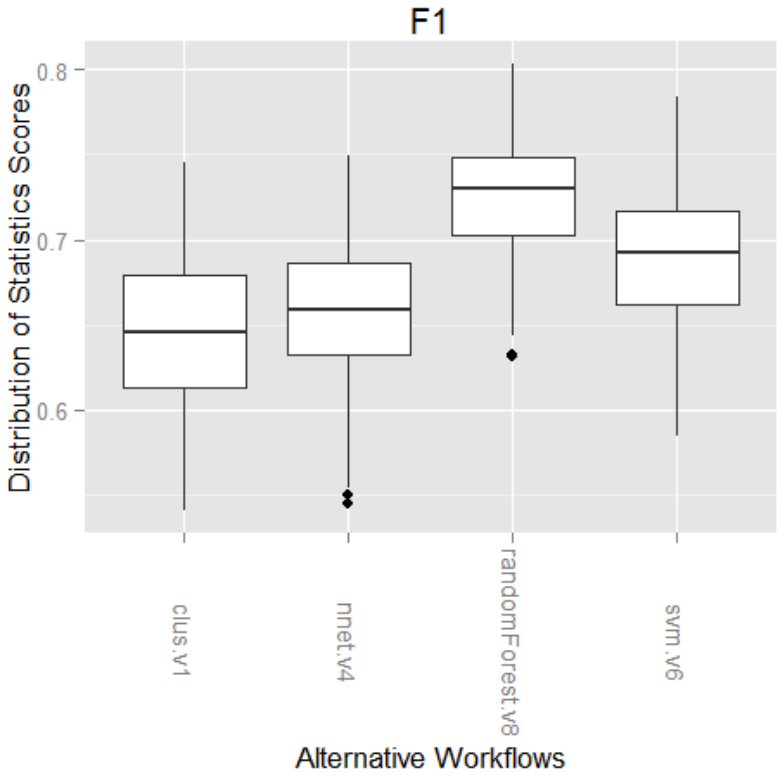

Figure 9: F1 measure of the models in the experiments.

6.2 Sentiment Scoring Results

For the task of tagging the documents for sentiment we have considered each topic individually (c.f. Section 4.3). In this context, we follow the same strategy in the presentation of the results of the models we have considered.

As we have seen in Section 5.3, the Total Cost (c.f. Equation 9, page 18) is the metric that will be used to compare the different model variants. To serve as a kind of baseline performance in terms of cost, we have used a naive model that predicts as score for all documents the most frequent score in the training data. This naive strategy appears in the results tables under the label **ModePred**. Using this baseline score we also calculate the Relative Cost for each model, which is the ratio of its Total Cost over the Total Cost achieved by the baseline.

As we have mentioned in Section 4.3, we have considered two alternatives for the sentiment scoring task - regression and classification. To distinguish which alternative is using each model variant we have included an "r" at the end of its name if it is using a regression approach, whilst the classification approaches do not have this ending.

In analyzing the results we should remark that the used metrics (Total Cost and Relative Cost) are to be minimized, i.e. having lower values is better.

The next sub-sections present the results for each of the 3 topics that were considered in our experiments.

6.2.1 Photovoltaic Economic Aspect

Table 5 presents the results of the best variants (classification and regression) of each modeling technique we have considered. As we can observe the best overall score for this topic was obtained by a random forest using a classification approach. Compared to the baseline of always predicting the most common score, this best model did not achieve an impressive improvement (8%) . Moreover, we also observe that several models (notably the neural network variants) achieve a score that is worse than the baseline, which means they are useless. This type of behavior may indicate that most documents have the same sentiment score and thus it is hard to beat a model that always predict this most common score. In effect, if we take into account that the test set sizes within the 10×10-fold cross validation process are ≈ 50 documents ($1/10$ of the number of documents of this topic, c.f. Table 2), we can confirm that the naive model performs rather well because an average total cost of 50.80 means roughly a cost of ≈ 1 for each classified document and thus very accurate predictions if we take into account the cost matrix that was used (c.f. Table 3). In this context, it makes sense that the other modeling approaches have difficulties in clearly outperforming this naive strategy.

Concerning the issue on whether it is better to use a classification or regression approach, the results reveal that the best scores are obtained with the classification approaches, irrespectively of the modeling technique.

Figure 10 provides a more detailed analysis of the results by showing the distribution of the scores of these variants over the 100 repetitions through box plots.

Overall, and in spite not being able to clearly outperform the baseline for the reasons explained above, we would said that the results are good, because an average absolute deviation of slightly less than 1 on the sentiment score of each document is an interesting achievement. For instance, this means that a document with a positive sentiment on a topic (score > 0) will "never" be tagged as having a negative sentiment, or vice-versa.

Table 5 Best performing models for the photovoltaic economic aspect topic.

Model	Total Cost	Relative Cost
svm.v5	48.20±5.49	0.95
svmr.v21	49.10±3.28	0.97
randomForest.v5	**46.90±8.14**	**0.92**
randomForestr.v9	50.40±2.41	0.99
nnet.v18	53.40±4.01	1.05
nnetr.v2	66.30±5.77	1.31
ModePred	50.80±5.25	1.00

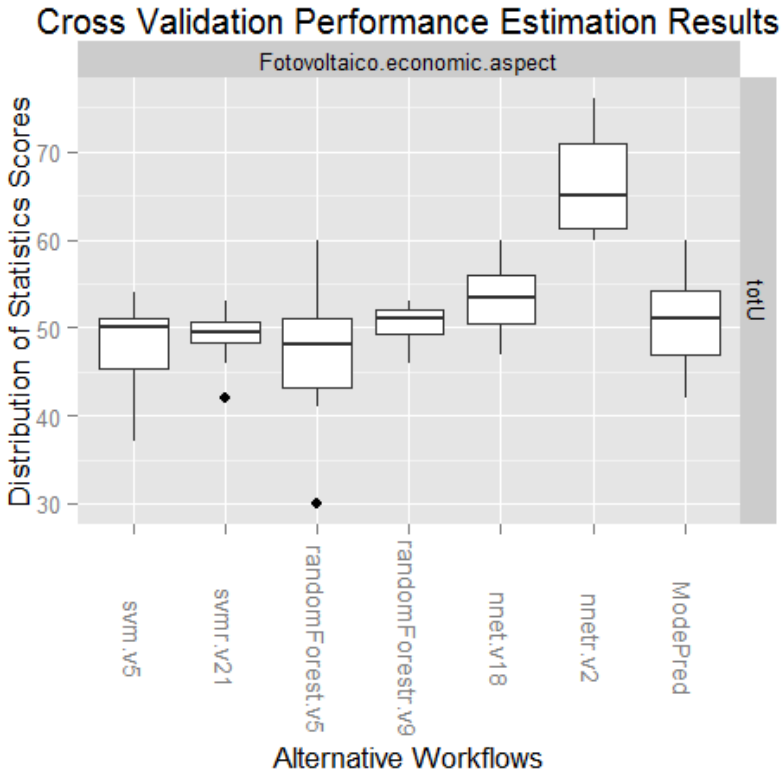

Figure 10: Distribution of the performance of the models in the experiments.

6.2.2 Photovoltaic Environmental Aspect

Table 6 shows the results of the best model variants for the Photovoltaic Environmental Aspects topic. In this topic we observe that the best results are achieved by SVMs, and in particular one that uses a classification approach. When compared to the naive baseline, in this topic the results are more interesting, as this baseline behaves considerably worse. With the exception of neural networks we have once again confirmed the advantage of classification approaches over the regression alternatives. Figure 11 provides information on the distribution of the scores across all repetitions of the experimental comparison.

Overall, given that the test sets of this topic have roughly 6 cases ($1/10$ of the 63 documents available in this topic, c.f. Table 2), an average total cost of 4.9 is an interesting result as it again means less than 1 in absolute deviation of the predicted score given the used cost matrix (Table 3). As mentioned on the previous section this is an interesting property. Moreover, we can confirm in Figure 11 that his is a stable performance, as only once in the 100 repetitions the score was above 6.

Table 6 Best performing models for the photovoltaic environmental aspect topic.

Model	Total Cost	Relative Cost
svm.v7	**4.90±2.18**	**0.57**
svmr.v19	5.60±1.17	0.65
randomForest.v7	5.50±2.32	0.64
randomForestr.v1	6.20±0.79	0.72
Mnnet.v5	5.99±1.86	0.70
Mnnetr.v3	5.83±1.41	0.68
ModePred	8.60±1.43	1.00

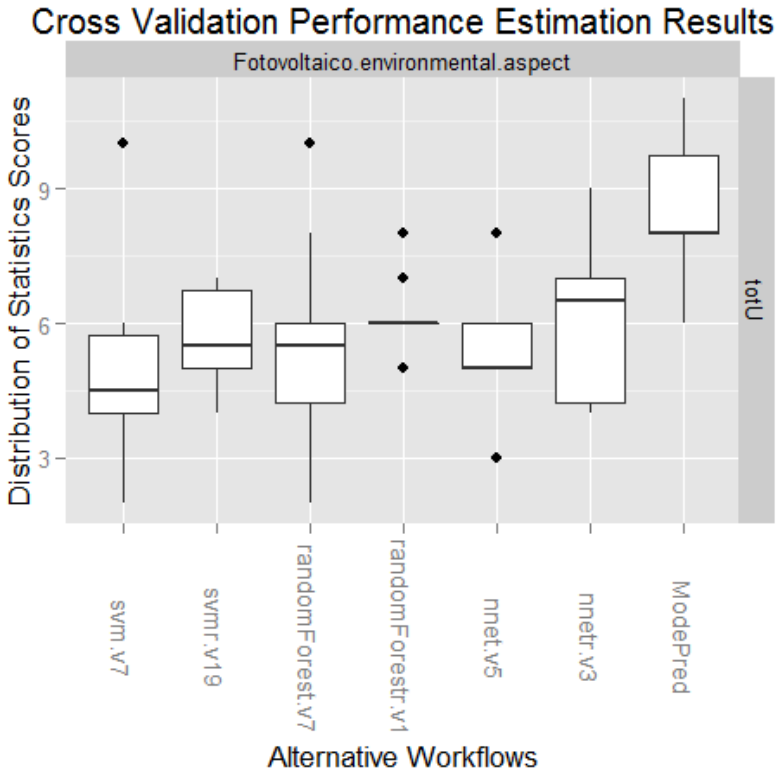

Figure 11: Performance of the models in the experiments.

6.2.3 Photovoltaic Technology Aspect

With respect to the results for the Photovoltaic Technological Aspects, Table 7 shows that random forests and SVMs achieve very similar performance with a slight advantage of the regression variant of SVMs. For this topic the test sets on the cross validation iterations have roughly 41 cases (1/10 of 410, c.f. Table 2) and thus the Total Cost achieved by the naive model (average of 31.90) is once again remarkable, and thus hard to beat, which can be confirmed by the relative cost scores achieved by our modeling approaches. Once again we show the distribution of the scores across all repetitions in Figure 12, which confirms the good performance of the best models by generally achieving an total cost that represents an average absolute deviation on the score of each document that is less than 1. In this context,

the results of these models can be considered very interesting.

Table 7 Best performing models for the photovoltaic technology aspect topic.

Model	Total Cost	Relative Cost
svm.v5	31.40±5.62	0.98
svmr.v18	**30.60±5.15**	**0.96**
randomForest.v11	30.60±7.14	0.96
randomForestr.v11	34.20±4.54	1.07
nnet.v9	34.00±5.79	1.07
nnetr.v23	40.70±4.67	1.28
ModePred	31.90±7.05	1.00

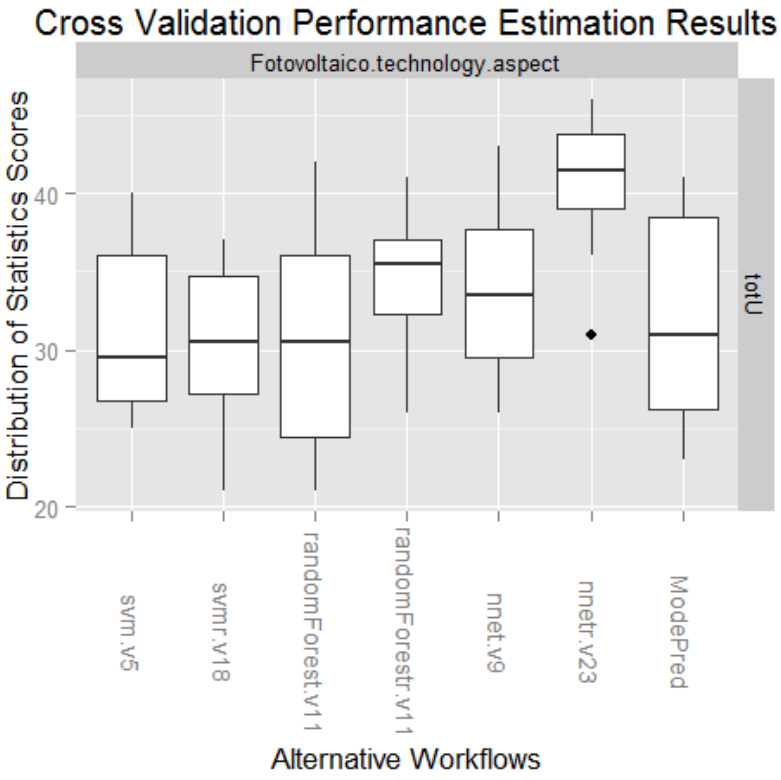

Figure 12: Performance of the models in the experiments.

7 Conclusions

This deliverable has described the second and last version of the evaluation of the opinion mining software prototype. This current evaluation is a complement of the initial evaluation described in Deliverable 6.3. We have focused the evaluation on two key components of this prototype: (i) identification of the topics that are mentioned in the posts, and (ii) scoring the expressed opinions in terms of sentiment. Having models that are able to perform well on these two tasks means that the opinion mining software prototype will be able to accurately infer the sentiment of the population concerning the set of predefined topics.

We have formalized these two predictive tasks that are involved in the general task of inferring the sentiment of the population concerning a set of energy-related topics. We have described a series of approaches to these tasks and their implementation in our prototype. We have proposed a series of performance metrics and the associated experimental methodology for the evaluation these key components of the OM prototype.

The results of our experimental evaluation indicate that in general our models are able to perform these two tasks remarkably well. In effect, the scores in terms of topic identification are interesting even-though space for improvements still exist, which may be a result of the small amount of tagged posts that was available to train the models. In terms of sentiment scoring the results are also interesting but again with space for improvement. In both tasks, the obvious way to try to improve our results is to obtain more tagged data, which is a time consuming task requiring large resources of human specialists. Still, the feedback we have obtained from human experts that have checked some of the predictions made by our prototype, is very positive, which means that the OM prototype has potential in terms of helping policy makers to get a better intuition on the sentiment of the population concerning energy-related policies.

A Model Variants

In this appendix we describe the variants of the models detailing the parameter values that were used in each variant. The models whose name ends in *r* are the variants in which regression was used to obtain the results.

Table 8 Random Forests parameters.

Name	Number of trees	Mtry
randomForest.v1	100	3
randomForest.v2	200	3
randomForest.v3	500	3
randomForest.v4	1000	3
randomForest.v5	100	5
randomForest.v6	200	5
randomForest.v7	500	5
randomForest.v8	1000	5
randomForest.v9	100	7
randomForest.v10	200	7
randomForest.v11	500	7
randomForest.v12	1000	7
randomForestr.v1	100	3
randomForestr.v2	200	3
randomForestr.v3	500	3
randomForestr.v4	1000	3
randomForestr.v5	100	5
randomForestr.v6	200	5
randomForestr.v7	500	5
randomForestr.v8	1000	5
randomForestr.v9	100	7
randomForestr.v10	200	7
randomForestr.v11	500	7
randomForestr.v12 1000	7	

Table 9 Support Vector Machines parameters.

Name	Cost	Epsilon	Gamma
svm.v1	3	0.1	0.1
svm.v2	5	0.1	0.1
svm.v3	7	0.1	0.1
svm.v4	9	0.1	0.1
svm.v5	3	0.1	0.01
svm.v6	5	0.1	0.01
svm.v7	7	0.1	0.01
svm.v8	9	0.1	0.01
svm.v9	3	0.1	0.05
svm.v10	5	0.1	0.05
svm.v11	7	0.1	0.05
svm.v12	9	0.1	0.05
svm.v13	3	0.1	0.05
svm.v14	5	0.01	0.1
svm.v15	7	0.01	0.1
svm.v16	9	0.01	0.1
svm.v17	3	0.01	0.01
svm.v18	5	0.01	0.01
svm.v19	7	0.01	0.01
svm.v20	9	0.01	0.01
svm.v21	3	0.01	0.05
svm.v22	5	0.01	0.05
svm.v23	7	0.01	0.05
svm.v24	9	0.01	0.05
svm.v25	3	0.05	0.1
svm.v26	5	0.05	0.1
svm.v27	7	0.05	0.1
svm.v28	9	0.05	0.1
svm.v29	3	0.05	0.01
svm.v30	5	0.05	0.01
svm.v31	7	0.05	0.01
svm.v32	9	0.05	0.01
svm.v33	3	0.05	0.05
svm.v34	5	0.05	0.05
svm.v35	7	0.05	0.05
svm.v36	9	0.05	0.05

Table 10 Support Vector Machines parameters.

Name	Cost	Epsilon	Gamma
svmr.v1	3	0.1	0.1
svmr.v2	5	0.1	0.1
svmr.v3	7	0.1	0.1
svmr.v4	9	0.1	0.1
svmr.v5	3	0.1	0.01
svmr.v6	5	0.1	0.01
svmr.v7	7	0.1	0.01
svmr.v8	9	0.1	0.01
svmr.v9	3	0.1	0.05
svmr.v10	5	0.1	0.05
svmr.v11	7	0.1	0.05
svmr.v12	9	0.1	0.05
svmr.v13	3	0.1	0.05
svmr.v14	5	0.01	0.1
svmr.v15	7	0.01	0.1
svmr.v16	9	0.01	0.1
svmr.v17	3	0.01	0.01
svmr.v18	5	0.01	0.01
svmr.v19	7	0.01	0.01
svmr.v20	9	0.01	0.01
svmr.v21	3	0.01	0.05
svmr.v22	5	0.01	0.05
svmr.v23	7	0.01	0.05
svmr.v24	9	0.01	0.05
svmr.v25	3	0.05	0.1
svmr.v26	5	0.05	0.1
svmr.v27	7	0.05	0.1
svmr.v28	9	0.05	0.1
svmr.v29	3	0.05	0.01
svmr.v30	5	0.05	0.01
svmr.v31	7	0.05	0.01
svmr.v32	9	0.05	0.01
svmr.v33	3	0.05	0.05
svmr.v34	5	0.05	0.05
svmr.v35	7	0.05	0.05
svmr.v36	9	0.05	0.05

Table 11 Neural Networks parameters.

Name	Size	Decay	Max Iterations
nnet.v1	3	0.1	1000
nnet.v2	5	0.1	1000
nnet.v3	7	0.1	1000
nnet.v4	9	0.1	1000
nnet.v5	3	0.01	1000
nnet.v6	5	0.01	1000
nnet.v7	7	0.01	1000
nnet.v8	9	0.01	1000
nnet.v9	3	0.05	1000
nnet.v10	5	0.05	1000
nnet.v11	7	0.05	1000
nnet.v12	9	0.05	1000
nnet.v13	3	0.1	2000
nnet.v14	5	0.1	2000
nnet.v15	7	0.1	2000
nnet.v16	9	0.1	2000
nnet.v17	3	0.01	2000
nnet.v18	5	0.01	2000
nnet.v19	7	0.01	2000
nnet.v20	9	0.01	2000
nnet.v21	3	0.05	2000
nnet.v22	5	0.05	2000
nnet.v23	7	0.05	2000
nnet.v24	9	0.05	2000
nnetr.v1	3	0.1	1000
nnetr.v2	5	0.1	1000
nnetr.v3	7	0.1	1000
nnetr.v4	9	0.1	1000
nnetr.v5	3	0.01	1000
nnetr.v6	5	0.01	1000
nnetr.v7	7	0.01	1000
nnetr.v8	9	0.01	1000
nnetr.v9	3	0.05	1000
nnetr.v10	5	0.05	1000
nnetr.v11	7	0.05	1000
nnetr.v12	9	0.05	1000
nnetr.v13	3	0.1	2000
nnetr.v14	5	0.1	2000
nnetr.v15	7	0.1	2000
nnetr.v16	9	0.1	2000
nnetr.v17	3	0.01	2000
nnetr.v18	5	0.01	2000
nnetr.v19	7	0.01	2000
nnetr.v20	9	0.01	2000
nnetr.v21	3	0.05	2000
nnetr.v22	5	0.05	2000
nnetr.v23	7	0.05	2000
nnetr.v24	9	0.05	2000

Table 12 Clus parameters.

Name	Ensemble Type
clus.v1	Bagging
clus.v2	Random Forest
clus.v3	Random Subspaces
clus.v4	Bagging of Subspaces

References

[1] Clus framework. URL: http://dtai.cs.kuleuven.be/clus/.

[2] C. Banea, R. Mihalcea, and J. Wiebe. Multilingual sentiment and subjectivity analysis. *Multilingual Natural Language Processing*, 2011.

[3] Hendrik Blockeel, Luc De Raedt, and Jan Ramon. Top-down induction of clustering trees. pages 55–63, 1998. URL: http://www.cs.kuleuven.ac.be/cgi-bin-dtai/publ_info.pl?id=20419.

[4] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20(3):273–297, 1995.

[6] Nello Cristianini and John Shawe-Taylor. *An introduction to support vector machines and other kernel-based learning methods*. Cambridge university press, 2000.

[7] Energetica Ambiente Forum. `http://www.energeticambiente.it/index.php`.

[8] Newclear blog. `http://blog.forumnucleare.it/`.

[9] Andy Liaw and Matthew Wiener. Classification and regression by randomforest. *R News*, 2(3):18–22, 2002.

[10] G. Madjarov, D. Kocev, D. Gjorgjevikj, and S. DÅ¿eroski. An extensive experimental comparison of methods for multi-label learning. *Pattern Recognition*, 45(9):3084–3104, 2012.

[11] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The Bulletin of Mathematical Biophysics*, 5(4):115–133, 1943.

[12] David Meyer. Package e1071. *R News*, 2012.

[13] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Cognitive modeling*, 1:213, 2002.

[14] W. N. Venables and B. D. Ripley. *Modern Applied Statistics with S*. Springer, New York, fourth edition, 2002. ISBN 0-387-95457-0.

[15] B Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2004.