# An approach to improve the accuracy of probabilistic classifiers for decision support systems in sentiment analysis

Vicente García-Díaz[a], Jordán Pascual Espada[a], Rubén González Crespo[b], B. Cristina Pelayo G-Bustelo[a], Juan Manuel Cueva Lovelle[a]

[a] *University of Oviedo, Department of Computer Science, Sciences Building, Oviedo, Asturias, Spain*
[b] *Universidad Internacional de La Rioja (UNIR), Av. de la Paz, 137, 26006 Logroño, La Rioja, Spain*

**Abstract**

Social networks link people and machines, providing a huge amount of information that grows very fast without the possibility to be handled manually. Moreover, opinion mining is the process of using natural language processing, text analytics and computational linguistics to identify and extract subjective information in different sources such as social networks. To that, classification methods are used but due to the limitless number of topics and the breadth and ambiguity of natural language, with its peculiarities in social networks, the results can be greatly improved. In this work, we present DSociaL, a platform to automate the processing of information obtained from social networks, focusing on improving the accuracy of decision support systems for sentiment analysis. We focus on machine learning-based simple probabilistic classifiers, evaluating a naive Bayes classifier, the basis of one of the most used soft computing techniques. Thus, we show a use case in which the proposal, with definitions and refinements made by experts, helps to improve the prediction of users' feelings towards a movie compared to what would happen with a conventional approach.

*Keywords:* Sentiment analysis, Classifier, Bayesian Network, Soft Computing, Domain-Specific Language, Machine learning

## 1. Introduction

Networks are groups of interrelated or interconnected elements, especially over large areas. Thus, networks of people exist from the origin of the human race. They are very important because those groups of people bound together may undertake an objective with greater ease. There are different types of groups such as informal networks, project teams, formal work groups or communities of practice, all of them composed under different circumstances and useful in complementary ways [1].

*Email addresses:* garciavicente@uniovi.es (Vicente García-Díaz), pascualjordan@uniovi.es (Jordán Pascual Espada), ruben.gonzalez@unir.net (Rubén González Crespo), crispelayo@uniovi.es (B. Cristina Pelayo G-Bustelo), cueva@uniovi.es (Juan Manuel Cueva Lovelle)

In addition, when computers link people and machines, they become social networks. Social Network Sites (SNS) such as Facebook, Twitter, Google+ or LinkedIn have attracted millions of users that are using those sites in their daily practices [2]. More than a billion of people in the world is connected together to create, collaborate and contribute their knowledge and wisdom, having social related factors the most significant impact on the intention of use [3]. In this regard, there are many works that use the current capacity of social networks, where the geo-localization plays an important role, to perform different types of research [4, 5, 6, 7].

Microblogging is a form of social communication in which users describe their feelings in short messages distributed through the Internet by using Web, desktop or mobile applications. People use such services mainly to talk about their daily activities and to look for or share any type of information [8]. For example, Twitter is a microblogging service that basically allows users to write about any topic within the 140-character limit and follow others to receive their messages (called *tweets*). It has some features such as non-power-law follower distribution, a short effective diameter and low reciprocity [9], which determines a differentiation from other existing non-virtual networks [10].

With the evolution of social networks, there is available a great amount of digital content and shared knowledge resources that produce environments with a mixture between the architecture of the underlying information and the social structure of the groups of people that are part of the communities [11]. This has lead to the appearance of a huge amount of data that is impossible to be manually treated by people as it grows in millions of bytes every day. In fact, it has been estimated that the amount of data stored in databases in the world doubles every 20 months [12].

The Knowledge Discovery in Databases (KDD) process deals with the huge amount of available data with the aim of discovering new knowledge from the information that is stored in different systems. It has usually five steps [13]: 1- selection; 2- pre-processing; 3- transformation; 4- data mining; and 5- interpretation. Data mining is an interdisciplinary field with lots of applications that refers to the process of discovering patterns in large data sets by the use of other fields such artificial intelligence, statistics or programming algorithms [14]. Thus, text mining or text data mining [15] is the process of deriving high-quality information from written texts. Moreover, sentiment analysis (a.k.a. as opinion mining) [16] is one of the most interesting applications of text mining. We will focus on opinion mining in this work. It refers to the use of Natural Language Processing (NLP) techniques, together with text analytics and computational linguistics to identify and extract subjective information from texts.

To deal with opinion mining related problems, soft computing techniques (a.k.a. computational intelligence) are usually used. It consists in the use of inexact solutions to computationally difficult tasks, for which there is no known algorithm that gives correct solutions in a constraint amount of time. So, soft computing is tolerant to imprecision or uncertainty. In addition, machine learning plays an important role in soft computing. It is a subfield of computer science that gives computers the ability to learn without being explicitly programmed [17]. Machine learning is widely used to categorize anything for which there is no exact precision on how to deal with it. A clear example is sentiment classification.

In order to classify the sentiment of a text, classifiers are used. In machine learning and statistics, classification focuses on identifying to which category a new observation belongs on the basis of a training set of data that has observations, whose category is

known beforehand. Classification is considered part of supervised learning.

The most well-known type of classification is probabilistic classification. To that, statistical inference is needed to find the best class for a given observation, based on the probability for each of them. Naive Bayes classifier is the most common probabilistic classifier and refers to a family of simple classifiers based on applying Bayes theorem with strong independence assumptions among the different variables or features. Being able to know the characteristics of each of the classifiers, and how to configure them for all the possible cases is not a simple task and requires the knowledge of expert developers, among other factors [18].

Working with all the concepts, parameters, algorithms and tools available for sentiment analysis is not a trivial task and requires the use of expert developers. Thus, Domain-Specific Languages (DSLs) can play an important role. According to Walton [19], a DSL is a small, usually declarative, language expressive over the distinguishing characteristics of a set of programs in a particular problem domain. Those little languages, tailored towards the specific needs of a particular domain can significantly ease building software systems for such a domain [20]. Even, domain-experts can directly use the DSL to make required routine modifications [21] or even program applications for their domain of knowledge (e.g., Garcia-Diaz et al. [22] for building food traceability applications under a Model-Driven Engineering approach [23]).

Getting a good sentiment analysis is key to recommendation or feedback systems, in which the users sentiment towards a product or service can be decisive in the future actions of the provider or to other potential customers. Thus, the main goal of this research work is to provide a novel approach to improve the accuracy of probabilistic classifiers, which are applied in the field of sentiment analysis in messages related to current issues. For example, messages related to a new movie, social events, political decisions, etc. As the characteristics and expressions used in social networks have their particular peculiarities (e.g., colloquial expressions, abbreviations, emoticons, dynamism, etc.), we will focus on the fourth step in the KDD process for improving opinion mining accuracy on social networks.

Our approach allows experts in specific domains of knowledge to identify positive and negative features that current classifiers are not able to automatically identify as key elements, to correctly classify the polarity of a given text. By using a DSL, even people without a background in traditional programming languages can use it to define rules and metarules that can improve the performance of a given classifier.

This approach is based on a post-processing task performed using the cosine similarity between the analyzed messages and a set of text snippets and logical expressions. This set could be, for example, few phrases or expressions related to a current event. The output modifies directly the probability for a message to be positive, negative o neutral. So, this solution can be used after the machine learning algorithm for detecting the message polarization.

The remainder of this work is structured as follows: In Section 2 we present a background with advances of the state of the art related to sentiment analysis, focusing on social networks. In Section 3 we propose our alternative to improve access to data in social networks focusing on getting the sentiment of a message. Section 4 shows an evaluation of the proposal. Finally, Section 5 deals with the conclusions and future work to be done.

## 2. Background

A wide range of research work is focused on the sentiment classification [24, 25, 26]. There are lot of different approaches, being very common the division into two well differentiated types of methods [27]:

- Lexicon based methods. There are a lot of methods used in some research works such as the pointed out by Taboada et al. [28]. These techniques are mainly based on dictionaries of words annotated with their semantic polarity.
- Machine learning based methods. They are divided also in groups: super-vised and unsupervised techniques [29]. In addition, some authors mention a hybrid between these both: semi-supervised learning [30, 31]. These techniques are typically more complex and, in the most usual case, require to create a model by training a classifier with labeled examples.

The different works rely on NLP. For example, Alonso et al. [32], propose a linguistic consensus model for Web communities, with some feedback mechanisms to improve speed and convergence. Godbole et al. [33] work on the sentiment analysis of online news and blogs texts, assigning scores to each distinct entity in the text corpus. Other works such as Gonzalez et al. [34], also explore the use of Twitter. They propose a novel approach to obtain a fine grain sentiment analysis with semantics.

Many polarity analysis use feature selection to classify the message polarity of the text or some snippets of it [35]. These approaches extract key parts of the message like nouns, verbs and adjectives to create n-grams. These keywords are used as features in machine learning algorithms like Naive Bayes, which are a family of algorithms commonly used in messages polarity classification [36].

Many authors have used techniques based on analyze sets of n-grams, like bigrams and trigrams to consider consecutive words, because these sets could have a different meaning than every single word [37, 28]. Other approaches have used dependency trees to model the correlation between sets of words [38]. But there are other approaches which have achieved an improvement in the analysis results. Xia et al. have successfully introduced an ensemble technique to represent various features [39].

Information gain algorithms allow to measure the difference in text classification results when individual features are used, so these algorithms can be helpful to improve the accuracy of the classification [40]. These kind of models are useful in a lot of situations to improve the effective of the classification system, but its effectiveness continues been highly dependent on the message language and the contextual interpretation.

The polarity detection of messages is, in a lot of environments, a high complex process, due to the natural language ambiguity and even more in short and informal texts. There are more specific complexities when the messages are in a temporal or social context and the topics evolve in real time. It means the message could include new expressions related to trends and current social references which are difficult to analyze but there have a high sentimental load [41].

Beside these approaches are useful to improve the accuracy, there are not able to analyze many messages when the message contains a lot of references about a specific topic. So, there are some proposals in which the topic of the messages is an important factor of the analysis process. Yoon et al. apply combinations of regression models focused

on specifics topics [42]. Bollegala et al. propose a sensitive distributional thesaurus, using labeled data for the source domains and unlabeled data for both source and target domains [43]. Definitely, focusing on the analysis model in a specific topic improves the accuracy of results in comparison to generic models. And that is a very applicable solution, because in a large number of cases, the analysis systems are focused on very specific topics: current news or event, a politician party, a movie, etc.

Problems arise when messages start to include a lot of noise or phrases which are not taken into account in the analysis, for example specific phrases or expressions related to a current trend or context [44]. Thus, the accuracy of the system can start to decrease, since it is complex to adapt the sentiment analysis dynamically, because it needs training, and analyzing again the same messages with the new model could take a lot of time and lots of topics need dynamic and real time analysis. For example, when the system is analyzing messages about a live sports broadcasts.

Pre-processing could be a good method for normalizing messages or removing the noise before applying the sentiment analysis [45]. In most cases, these methods are used for removing or changing some parts of the messages [46]. Pre-processing processes can be simple or very complex [47] and are a great help, since they can be used in a non-intrusively and independent way. These pre-processing systems can be used always, for all messages, or eventually. They depend on the current trends or the dynamic evolution of the messages topic. Pre-processing can also be used for filtering, increasing or reducing the importance of some concepts.

But pre-processing techniques may not too useful to analyze messages about dynamic topics. If we detect the system is not analyzing the polarity of the messages in a good way, we can try to improve the detection dynamically, without changing the sentiment analysis algorithm applying pre-processing [48]. Designing the pre-processing system could not be simple, and we need to analyze the results to see if the pre-processing process is really improving the accuracy of the system. In a lot of cases, the design of the pre-processing process will need few adjustments. Every modification in the pre-processing process implies to analyze again the messages, especially if we do not want to lose this valuable information. This is so because the pre-processing processes do not modify directly the sentiment analysis result, they just focus on getting the messages in a more suitable state for the subsequent analysis [49, 50].

In the next section, we propose a novel approach to improve dynamically the results of the sentiment analysis when the results are being harmed by specific phrases or expressions related to a current trend or social reference. This approach is based on a post-processing analysis using the cosine similarity between the analyzed messages and a set of text snippets and logical expressions.

## 3. Proposal

In this section we are going to detail our proposal, DSociaL. First, we will explain the architecture of the whole system and then we will talk about the modified probabilistic model we integrate in the system to help to infer correct decisions through the use of a trained classifier.

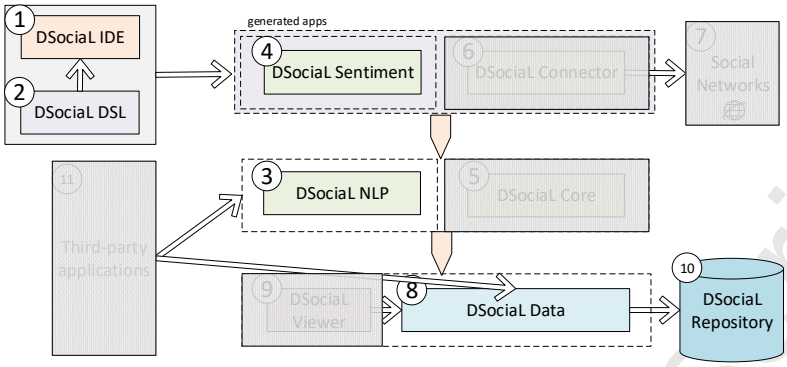


Figure 1: DSociaL overview

### *3.1. Components*

Fig. 1 shows an overview of the architecture and its components, that are explained below. Roughly, DSociaL is a platform that provides a language from which users can define which information they want to extract from social networks, store it in a repository, and perform natural language processing from the data that was previously stored.

#### *3.1.1. DSociaL DSL*

Most of the developers think of a programming language as one capable of programming any application with always the same degree of expressiveness and efficiency, that is, a General-Purpose Language (GPL). However, there are better and more natural ways to create solutions for problems tailored to a particular application domain by designing and creating DSLs that capture just the required semantics of the domain. That way, complete application programs can be developed more quickly and effectively than with a GPL [51]. In this work, we choose to design and use a DSL since there is a lack in this regard in the scientific literature from the point of view of redefining and extending classifiers behavior. In addition, the DSL approach is the way to link together a solution to a domain problem, that has been used for a long time with a large number of related works [52].

There are mainly two reasons why it is worth creating a new DSL [53]: 1) improved software economics; and 2) allow that people with less domain and programming expertise to develop software, even end-users with some domain, but no programming expertise [54]. Since our goals are compatible with both criteria, we have created a new DSL based on common elements used to describe different factors that can help to adjust the behavior of a classifier with expert knowledge. The same ideas have been applied to many works, some of them related to machine learning problems [55]. To that end, we have used the Xtext framework [56]. From a grammar and some other definitions, it is possible, for example, to get a working parser and linker and also a complete Eclipse-based Integrated Development Environment. Xtext also provides several mechanisms through which you can configure different aspects of languages such as validations of code, syntax highlighting, proposals to developers, code formatting or even generating

artifacts through programs implemented with the programming languages defined with Xtext.

Table 1 summarizes the main features of the language to enrich the results obtained by using any type of sentiment classifier. Thus, to achieve compliance with requirements, we have defined a meta-model, whose image is partially represented in Fig. 2. The idea is similar to the feature descriptions presented by van Deursen and Klint [57]. Thus, every time a new program with DSociaL DSL is created, a model that conforms to the proposed metamodel is instantiated, following its rules and offering a formalism that makes it very easy to perform different tasks such as validation, storing or generation of artifacts. The rules are defined in general terms based on the metamodel of the language, not for each individual case, that is, not for each model obtained during the development, which facilitates the process.

| Id | Name | Description |
|---|---|---|
| 1 | Positive constraints | They are used to highlight the positivity of expressions found in a text |
| 2 | Negative constraints | They are used to highlight the negativity of expressions found in a text |
| 3 | Neutral constraints | They are used to detect neutral expressions in a text |
| 4 | Weights of constraints | Positive and negative constraints have a weight (from 0 to 1) to adjust the result obtained by the classifier that has been trained with an specific train set |
| 5 | Similarities | Expressions used by constraints have a cosine similarity measure (from 0 to 1; 1 by default) that is used to match texts that, although similar, are not necessarily the same as the indicated expression. A value of 1 means that they should be identical |
| 6 | Metarules | They are used to combine different constraints creating more complex expressions with logical operators, that also have a weight |
| 7 | Negative threshold | Sometimes, some rules can penalize too much the final result. This threshold limits the total possible negativity |
| 8 | Positive threshold | Sometimes, some rules can potentiate too much the final result. This threshold limits the total possible positivity |

Table 1: Main features of the language

*3.1.2. DSociaL IDE*

Based on the Xtext architecture, some of the features included in the DSociaL IDE development environment are:

- Custom syntax-highlighting to distinguish the different elements of the language (e.g., keywords, comments or variables).

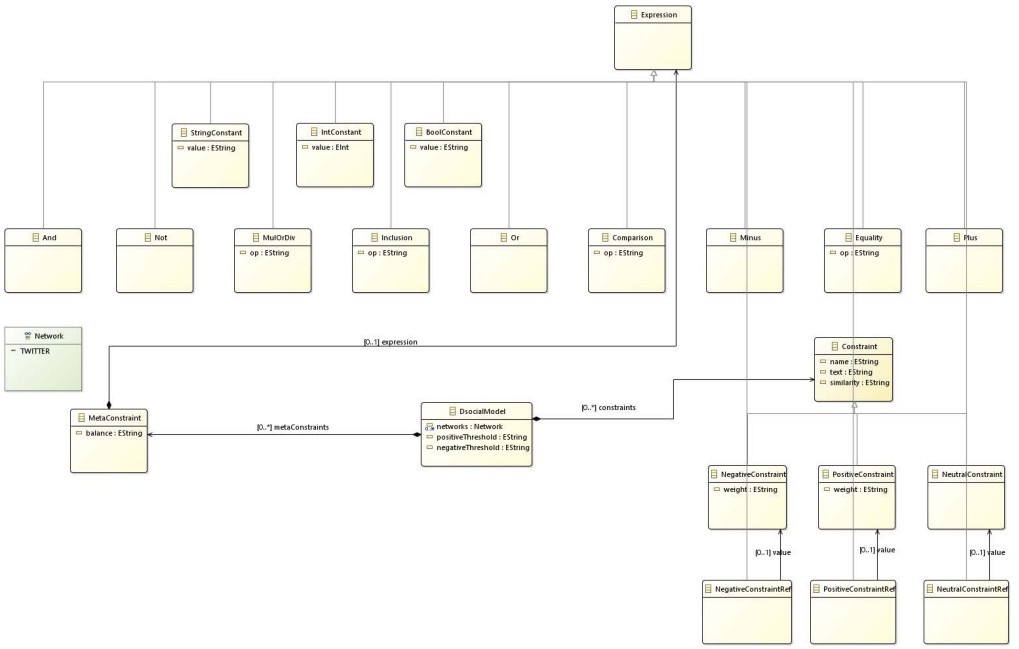


Figure 2: Metamodel fragment of the DSL

- Content assistant to help the developer to write code faster and more efficiently through the use of the auto-complete functionality.
- Static validation of the language elements to detect syntactic and semantic issues.
- Suggestions for fixing errors or problems identified in the code.
- Templates that allow developers to reduce the learning curve for typical operations.
- Formatting the code through a feature called code beautifier to distribute it properly and promote its maintenance.
- Outline view fully configurable to both the elements that appear and text or icons attached to them.

Fig. 3 is a screenshot of the environment when a model is being created. It can be seen different features. For example, the syntax-highlighting for different elements (e.g., `networks`, `constraints`, `ignore`, etc.), a static validation error marking a `balance` as not valid because instead of a float number, the programmer typed a string value, and the outline view showing a summary of the elements that are being used (in the example just two positive constrains, one negative constraint, two neutral constraints and two meta rules).

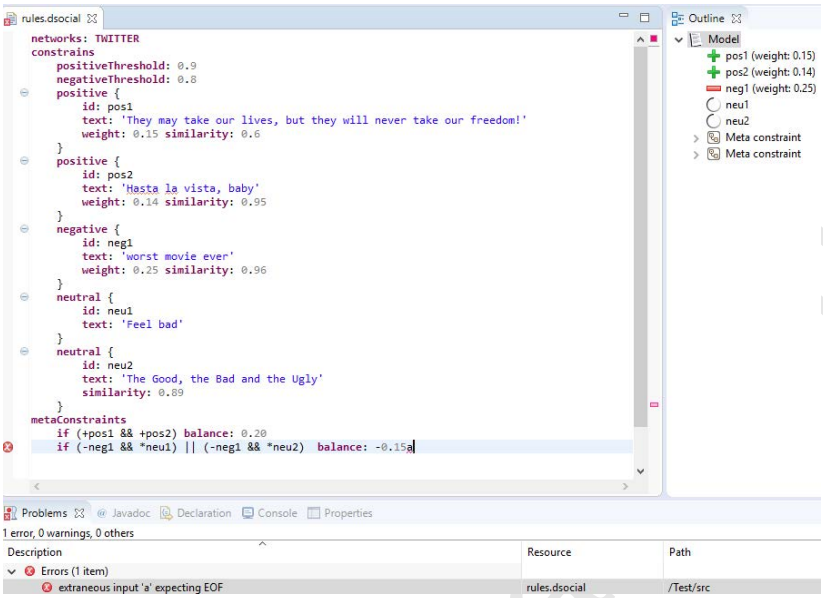


Figure 3: DSociaL IDE sreenshot

### *3.1.3. DSociaL NLP*

This component is responsible for performing different tasks related to natural language processing. For example, it allows to access data that is stored in the system repository and trains them to classify the sentiment of people who write about a given product as positive or negative, depending on whether people speak favorably or not about the product.

Focusing on the sentiment analysis, it performs some operations such as data preprocessing with tokenization and normalization. Some common operations (e.g., [58, 59]), we carry out are the following:

- Emoticons are taken into account, since there are very common in social networks [60]. Thus, :-) or :-( express something positive or negative respectively
- All capital letters are replaced by lower case letters.
- The presence of special elements and symbols (e.g., URLs, user names, commas) are substituted.
- Additional white spaces are removed because they do not provide semantic information.
- Hashtags symbols (#) are removed from the words they precede in order to be part of the text.
- Informal intensifiers and character repetitions are also identifed.

- A list of `stopwords` is also used to remove very common words that can reduce the performance of the classifier.

To perform different analysis we work with different `n-grams` approaches [61]. There are contiguous sequences of `n` words from a given sequence of text. In DSociaL NLP we define 1) unigrams; 2) bigrams; 3) trigrams; 4) unigrams and bigrams; 5) bigrams and trigrams; and 6) unigrams, bigrams and trigrams, all together.

*3.1.4. DSociaL Sentiment*

Based on the DSociaL NLP infrastructure, we can generate applications on the basis of the definitions made by the users of the DSL. For example, DSociaL Sentiment is a layer of abstraction on top of the DSociaL NLP that is used to calculate the sentiment of the sender of a message using a modified probabilistic model. Both components (DSociaL NLP and DSociaL Sentiment) are created using the Python programming language, due to the wealth of frameworks that exist to work with natural language processing and machine learning such as NLTK [62]. For example, with a definition made using the DSL like in Listing 1:

Listing 1: Example of use of the DSL

```
positive {
       id: pos2
       text: 'Hasta la vista, baby'
       weight: 0.14 similarity: 0.95
}
```

We will internally generate the fragment like in Listing 2 of Python code in the application, that will call different already defined functions such as `are_similar` or `set_balance`, all transparently to the end user. However, we could generate code for any other platform the same way without changing anything from the point of view of the final user.

Listing 2: Snippet of Python-generated code

```
pos2 = similarity.are_similar('Hasta la
vista, baby', text, 0.95)

if (pos2):
       self.set_balance(0.14)
```

*3.1.5. DSociaL Data*

This is the component that stores the data that is obtained from different social networks. It is based on a RESTful API [63] to provide an interface to receive and store data in the repository or to access and distribute data from the repository. At this point, we can manage any data provided by the Twitter API using different services, for future analysis and processing. Some of them are `/message`, `/directMessage`, `/liked`, `/follow`, `/place`, among others.

*3.1.6. DSociaL Repository*

We store the data in a repository, that is a NoSQL database (MongoDB in our case). Thus, we store and retrieve data using documents in collections instead of tabular relations in traditional relational databases. There are mainly some performance and flexibility reasons why it could be interesting to use a NoSQL database, although there are not always the best option [64]. For us, simplicity of design, horizontal scaling and flexibility are important factors to take into account given the large amount of data that is handled in social networks. For example, messages from social networks other than Twitter could have different fields, but with a NoSQL database, that will not require to modify the schema of the database since it is flexible enough to be adapted.

*3.1.7. Other components*

Although not used in this work, the system has other components that can be used for different purposes. For example, DSociaL Core provides different basic types and algorithms that can be common to access different social networks. DSociaL Connector is an application that can be generated to extract data from social networks based on the indications made by the user. At this point we are only working with Twitter but we plan to work with other social networks in the future. With DsociaL Viewer we have a Web-based interface to show the data that is stored in the main repository and finally, using DSociaL NLP, DSociaL Core and the DSociaL Data we also plan to create new applications to provide further functionalities, and leave it open so others can do it too.

All the components were created with scalability and multiplatform execution in mind. For example, DSociaL DSL is based on Xtext, a framework that can work on any Java-based environment, including the Web. DSociaL NLP is composed of Web services, so it could be used or exported to any other platform or language in the future without any relevant changes in the system. The same occurs with the data base. Although we are using now a NoSQL environment (MongoDB), since all the communication is performed using Web services, we could change it in the future or move to other server without any relevant change. That allows the creation of third party applications with any programming language and for any platform without having to take into account technical decisions that have been taken in the development of the platform. Finally, since, we are working on test machines, execution times have not been taken into account at this moment. However, it is important to note that, for the end user, the proposal only adds a constant complexity O(1) on the time complexity needed without the decisions of the experts.

*3.2. Probabilistic models*

Unlike deterministic models, in which there is no uncertainty in either the inputs or the corresponding outpus of a model, probabilistic models deal with uncertainty, which usually happens in reality. Both are mathematical models to simulate real-life situations with equations to forecast the future.

Probabilistic models have random variables representing uncertain events that may occur, together with probability distributions to assign probabilities to the potential outcomes. For example, we know that a new message is going to be received but we do not have any idea of the content of such message, so by incorporating uncertainty in the model we can quantify risks in any process, leading to better decisions. Thus, we can have a range of potential outcomes instead of a single best guess.

Formally, a classifier is a function that assigns a class $\hat{y}$ to a sample $x$, where $x \in X$ and $y \in Y$, being $X$ the set of all the inputs and $Y$ the set of all the possible classes defined before training. In our case, $X$ is the collection of messages and $Y$ can be `positive` or `negative`, referring to the feeling of who wrote it towards a product or service.

Probabilistic classifiers predict, given a sample input, a probability distribution over a set of classes with a degree of certainty. For a given $x \in X$, they assign probabilities to all $y \in Y$ in a way that they have to sum one and $\hat{y} = arg - max_y P(Y = y|X)$

Thus, for any message $x$, we assume that its sentiment should be either positive or negative, so: $P(x_{positive}) + P(x_{negative}) = 1$

Some well-known examples of classifiers that work under those assumptions can be binomial regression, logistic regression, multilayer perceptrons or naive Bayes. The goal of this work is to make it easier to enhance the performance of any of the probabilistic models directly with knowledge from domain experts. Thereby, we include two new variables in the equation:

$$\hat{y} = arg - max_y P(Y = (y + \alpha + \beta)|X)$$
$$(P(x_{positive}) + \alpha) + (P(x_{negative}) + \beta) = 1$$

Where $\alpha$ and $\beta$ are two variables directly derived by definitions made by expert users. To calculate its values we need to do a double iteration, first adding positive values to $\alpha$ and calibrating $\beta$ and then adding negative values to $\beta$ and calibrating $\alpha$

We can define $A$ as the set of all positive features found by the expert where $a \in A$. We can also define $B$ as the set of all negative features found by the expert where $b \in B$. $w(c)$ is the weight of a specific feature where $c \in \{A, B\}$.

$$addPos = \begin{cases} \sum_{i=1}^{i=length(A)} w(a_i) & \text{if } \sum_{i=1}^{i=length(A)} w(a_i) < \gamma \\ \gamma & \text{otherwise} \end{cases} \quad (1)$$

$$subPos = \begin{cases} -\sum_{i=1}^{i=length(A)} w(a_i) & \text{if } \sum_{i=1}^{i=length(A)} w(a_i) < \gamma \\ -\gamma & \text{otherwise} \end{cases} \quad (2)$$

$$addNeg = \begin{cases} \sum_{i=1}^{i=length(B)} w(b_i) & \text{if } \sum_{i=1}^{i=length(B)} w(b_i) < \delta \\ \delta & \text{otherwise} \end{cases} \quad (3)$$

$$subNeg = \begin{cases} -\sum_{i=1}^{i=length(B)} w(b_i) & \text{if } \sum_{i=1}^{i=length(B)} w(b_i) < \delta \\ -\delta & \text{otherwise} \end{cases} \quad (4)$$

$\gamma$ and $\delta$ delimit the space of the solution, avoiding that either the positive or the negative features acquire too much weight in the final solution. Finally, we can determine the value for $\alpha$ and $\beta$ as follows:

$$\alpha = addPos + subNeg$$
$$\beta = addNeg + subPos$$

The previous equations fullfill that $\alpha + \beta = 0$. So, there is guarantee that the definitions made by the expert give as a result of the sum (taking into account $\alpha$ and $\beta$), a total of one, even when the expert will define the values as conditional and independent possibilities.

## 4. Evaluation

In order to test the system, we are going to use movie-related tweets, randomly obtained from Twitter in December 2016. To do this, Twitter was inspected manually trying to find comments about movies that were being broadcast in the cinema at that time (e.g., Bad Santa 2), as well as keywords that could lead to results on movies of various kinds (e.g., movie, film, cinema, etc.). Regarding the tweets, we collected a large amount of data, for future processing (e.g., id, urls, mentions, medias, hashtags, contributors, source, number of retweets, lang, number of likes, etc.). However, from the point of view of the processing, we just needed the text associated to the tweet. Then, three people have performed a manual check of all messages reaching different conclusions in consensus:

- The message has a negative connotation about a movie. It is marked as negative.
- The message has a positive connotation about a movie. It is marked as positive.
- The message has both negative and positive connotations. It is discarded.
- The connotation of the message about a movie is unclear. It is discarded.

Collected tweets were focused on the sentiment of users towards different movies, being the sentiment positive or negative. For this work, we have discarded the neutral feelings. This work differs from others and from the Twitter API in the sense of how we apply what is positive or negative. Thus, if a user says "I want to watch Superman again :(", we do not take it as a negative sentiment, but positive. This is so, because despite the user is showing his negativity, it seems pretty obvious that to him, Superman is a good movie. The same could be applied to products or any other type of service.

### *4.1. Naive Bayes classifier*

There are several different machine learning-based classifiers that can be used to solve a large number of problems. For example, Amos et al. [65] apply machine learning-based classifiers to dynamically detect Android malware. Petropoulos et al. [66] evaluate different classifiers for discriminating land-cover classes in the Mediterranean region, combined with hyperion hyperspectral imagery analysis [67]. Ramon-Pollan et al. [67] explores the design of mammography-based machine learning classifiers, focusing on breast cancer diagnosis. There are also some derived and interesting works such as Ateniese et al. [68], which deal with unexpected information that could be revealed from classifiers.

Naive Bayes classifiers [69] are used in lots of domains (e.g., in predition heart disease [70]). Moreover, there are widely used for text categorization or classification using word frequencies as features. It relies on the Bayes' theorem which assumes there is a naive independence assumptions among the different predictors. It is one of the most used, scalable and simple classifiers, but usually offers very good empirical results, outperforming more sophisticated classification algorithms. For example, Liu et al. [71] evaluate the scalability of Naive Bayes classifier in large datasets, showing that it scales better than other types of classifiers. In addition, some authors try to improve the performance of Naive Bayes classifiers. Thus, for example, Narayanan et al. [72] focus on a combination of different methods (e.g., negation handling, word n-grams, feature selection, etc.) to

improve the accuracy of Naive Bayes classifiers for sentiment analysis. Another interesting work is the one by Jiang et al. [73], that propose a locally weighted learning approach, that weakens the attribute conditional independence assumption made by Bayes. It can be compared with our approach in terms of the search of an improvement of the performance of the algorithm, avoiding assumptions that are not always true. However, our proposal relies more on the knowledge that an expert could have on a certain domain of knowledge, and can be applied to other classifiers.

Naive Bayes is a conditional probability model. Thus, given an instance to be classified with $n$ independent variables or features, represented by a vector $x = (x_1, ..., x_n)$, Bayes theorem provides a way to calculate the conditional probability:

$$P(c \mid x) = \frac{P(x \mid c) \times P(c)}{P(x)}$$

$$P(c \mid x) = P(x_1 \mid c) \times P(x_2 \mid c) \times ... \times P(x_n \mid c) \times P(c)$$

where:

- $P(c)$ is the prior probability of a specific `class`
- $P(x)$ is the prior probability of a specific `predictor`
- $P(x \mid c)$ is the probability of `predictor` given `class`, also known as likehood
- $P(c \mid x)$ is the probability of `class` given `predictor`

Focusing on our probabilistic model for the sentiment analysis we have:

$$P(positive \mid x) = \frac{P(x \mid positive) \times P(positive)}{P(x)} + \alpha$$

$$P(negative \mid x) = \frac{P(x \mid negative) \times P(negative)}{P(x)} + \beta$$

and we should also meet that:

$$P(positive \mid x) + P(negative \mid x) = 1$$

$$\alpha + \beta = 0$$

*4.2. Comparison*

We have trained the system using DSociaL NLP together with DSociaL Data, using a sample of 387 tweets, both with positive or negative feeling towards a movie. After preprocessing each of the messages, we have applied a Naive Bayes classifier to train them using k-fold cross-validation (with $k = 10$) and a final accuracy of $= 0.789$ using the training set. Of course, we could increase the number of messages and use different classifiers to improve the performance but for the purposes of our work, we did not find it necessary since our goal is to see how to improve the accuracy of a classifier using expert knowledge.

Table 2 shows some of the messages (from 1 to 19) for which the classifier has failed to indicate their correct feeling after training. There are different factors that affect the

| Id | Text |
|---|---|
| 1 | @bendreyfuss how is it glossed over? the entire point of the movie is about preventing that from taking place and they fail |
| 2 | RT @JonRisinger: It's funny that the best comic book villain in a movie so far is probably Samuel L. Jackson's character in Kingsman. |
| 3 | @Jonah_Heng try prison break breaking bad oth or arrow |
| 4 | deciding to re-watch breaking bad is a fabulous idea |
| 5 | I watched a #Nigerian movie today and i was disappointed. :( :( They said 25years later in the movie and the family Dog was still alive. |
| 6 | Bad Santa is the most hilarious movie I've ever |
| 7 | RT @pari_Sharma77: @Gurmeetramrahim Totally inspired movie I recommend to all must watch this movie #Celebrating50Days |
| 8 | @stillgray forget it, Jake. It's Chinatown |
| 9 | Just walked into a bar in which everyone was scream-singing the lyrics to "Dirty Little Secret". Chewie, we're home |
| 10 | RT @Vwanii: The magnificent seven ain't a bad movie. |
| 11 | It's torture to watch it tbh and I wouldnt recommend it unless it was ABSOLUTELY necessary. |
| 12 | #WheneverIFeelAfraid, afraid screams at me, "Get your stinking paws off me, you damned dirty ape!!" |
| 13 | RT @SupChardy: @vicegandako Your movie with Coco and the kids made my weekend super special.The laughter inside the cinema were all real!#T? |
| 14 | It is pathetic to watch it. I loss my favorite criminal series. |
| 15 | They may take our lives, but they'll never take our freedom!!! #ThingsIYellWhenICum |
| 16 | Rogue One was really fucking cool. Can definitely recommend. Felt more like a complete movie instead of a nostalgia montage. Watch it |
| 17 | @49ers @periscopeco Hasta la vista baby.. |
| 18 | Not bad.. the movie. Needed subtitles though ? |
| 19 | Fella today i have received your film TMNT 2, I saw it at the cinema last July, i liked it very much ?? https://t.co/gYD5n3ER0C |

Table 2: Messages with incorrect predictions after training the classifier

result. For example, in the text `Bad Santa is the most hilarious...` the fact that the word `Bad` is part of the title of a movie, can confuse the classifier, giving a negative factor to a movie that the user who wrote the message liked. Columns on Table 3 are as follows:

- `Column Id`. Id of the message related to Table 2.
- `Column Negative`. Probability that the feeling of the user towards the movie is negative based on the classifier.
- `Column Positive`. Probability that the feeling of the user towards the movie is positive based on the classifier.
- `Column Training`. Real sentiment of the user towards the movie.
- `Column Classifier`. Sentiment of the user towards the movie based on the classifier.
- `Column Negative (DSociaL)`. Probability that the feeling of the user towards the movie is negative based on our proposal.
- `Column Positive (DSociaL)`. Probability that the feeling of the user towards the movie is positive based on our proposal.
- `Column DSociaL`. Sentiment of the user towards the movie based on our proposal.

Table 3 shows the different results we obtained.

To obtain the DSociaL results, an expert in the domain, should create a model with this knowledge about it. Listing 3 shows it. Note that the values (thresholds, weights, similarities, texts and metarules) can change by offering totally different results. These values depend mainly on the users knowledge of the domain, and can be applied to any other domain. Note the use of regular expressions to increase the options matching different texts.

Listing 3: Model to enrich the classifier

```
networks: TWITTER
constrains
       positiveThreshold: 1
       negativeThreshold: 1
positive {
       id: pos1 text: '@\w+ try \w+' weight: 0.4
}
positive {
       id: pos2 text: '(.*) (re-)?watch (.*) fabulous (.*)' weight: 0.2
}
positive {
       id: pos3 text: 'is the most hilarious' weight: 0.5 similarity: 0.9
}
positive {
       id: pos4 text: 'must watch' weight: 0.3
}
```

| Id | Neg. | Pos. | Training | Classifier | Neg.(DSL) | Pos.(DSL) | DSociaL |
|---|---|---|---|---|---|---|---|
| 1 | 0.320 | 0.679 | negative | positive | 0.520 | 0.479 | negative |
| 2 | 0.692 | 0.307 | positive | negative | 0.692 | 0.307 | negative |
| 3 | 0.885 | 0.115 | positive | negative | 0.485 | 0.514 | positive |
| 4 | 0.566 | 0.534 | positive | negative | 0.365 | 0.634 | positive |
| 5 | 0.193 | 0.807 | negative | positive | 0.593 | 0.407 | negative |
| 6 | 0.936 | 0.064 | positive | negative | 0.565 | 0.435 | positive |
| 7 | 0.800 | 0.200 | positive | negative | 0.400 | 0.600 | positive |
| 8 | 0.502 | 0.498 | positive | negative | 0.202 | 0.798 | positive |
| 9 | 0.722 | 0.278 | positive | negative | 0.422 | 0.578 | positive |
| 10 | 0.973 | 0.027 | positive | negative | 0.673 | 0.327 | negative |
| 11 | 0.029 | 0.971 | negative | positive | 0.429 | 0.571 | positive |
| 12 | 0.896 | 0.104 | positive | negative | 0.596 | 0.404 | negative |
| 13 | 0.760 | 0.240 | positive | negative | 0.360 | 0.640 | positive |
| 14 | 0.0003 | 0.997 | negative | positive | 0.203 | 0.797 | negative |
| 15 | 0.995 | 0.005 | positive | negative | 0.695 | 0.305 | negative |
| 16 | 0.917 | 0.083 | positive | negative | 0.317 | 0.683 | positive |
| 17 | 0.520 | 0.479 | positive | negative | 0.220 | 0.779 | positive |
| 18 | 0.992 | 0.008 | positive | negative | 0.692 | 0.308 | negative |
| 19 | 0.718 | 0.282 | positive | negative | 0.418 | 0.581 | positive |

Table 3: Results obtained

```
positive {
       id: pos5 text: 'ain\'t a bad movie' weight: 0.3 similarity: 0.9
}
positive {
       id: pos6 text: 'super special' weight: 0.2 similarity: 0.9
}
positive {
       id: pos7 text: 'laughter' weight: 0.2 similarity: 0.9
}
positive {
       id: pos8 text: 'Watch it' weight: 0.3 similarity: 0.9
}
positive {
       id: pos9 text: '(.*) was really (.*) cool(.)?(.*)' weight: 0.3 similarity:
           0.9
}
positive {
       id: pos10 text: 'Not bad' weight: 0.3 similarity: 0.9
}
positive {
       id: pos11 text: 'liked it very much' weight: 0.3 similarity: 0.9
}
negative {
       id: neg1 text: 'they fail' weight: 0.2
}
```

```
negative {
       id: neg2 text: 'was dissapointed' weight: 0.2 similarity: 0.7
}
negative {
       id: neg3 text: 'torture to watch' weight: 0.2 similarity: 0.7
}
negative {
       id: neg4 text: 'wouldnt recommend' weight: 0.2 similarity: 0.7
}
negative {
       id: neg5 text: 'pathetic to watch' weight: 0.2 similarity: 0.7
}
neutral {
       id: neu1 text: 'watched'
}
neutral {
       id: neu2 text: 'I recommend'
}
neutral {
       id: neu3 text: 'forget it, Jake. It is Chinatown' similarity: 0.4
}
neutral {
       id: neu4 text: 'Chewie, we\'re home' similarity: 0.4
}
neutral {
       id: neu5 text: 'Get your stinking paws off me, you damned dirty ape'
           similarity: 0.4
}
neutral {
       id: neu6 text: 'They may take our lives, but they\'ll never take our
           freedom' similarity: 0.4
}
neutral {
       id: neu7 text: 'Hasta la vista baby' similarity: 0.5
}
metaConstraints
       if (*neu1 && -neg2) balance: -0.20
       if (*neu2 && +pos4) balance: 0.10
       if (*neu3 || *neu4 || *neu5 || *neu6 || *neu7) balance: 0.30
```

Finally, Fig. 4 and Fig. 5 shows the improvement obtained using DSociaL. Note that, when the optimal positive probability is one, the optimal negative probability should be zero, and vice versa, since the total probability should be one. In Fig. 4 the positive probabilities using DSociaL tend to be closer, than the results from the classifier, to the optimal values. Same applies for Fig. 5 with negative probabilities, where the results obtained with DSociaL tend to be closer to the optimal values. In such comparisons, we always compare the results from the original naive Bayes classifier with our proposal (DSociaL), with respect to the theoretical best result indicated by the people who have classified the messages manually as a positive or negative feeling toward a movie. So, the proposal, is an improvement over the original classifier. In this work we use Bayes because

it is one of the best known and used classifiers but we thought that the improvement would be equivalent to any other classifier or technique used, since the actions of the experts should allow to fine tune the results obtained by any method of classification. It is important to note that, our approach is an abstraction layer on top of other techniques, so the final idea is to improve any current of future technique, not replace any of them.

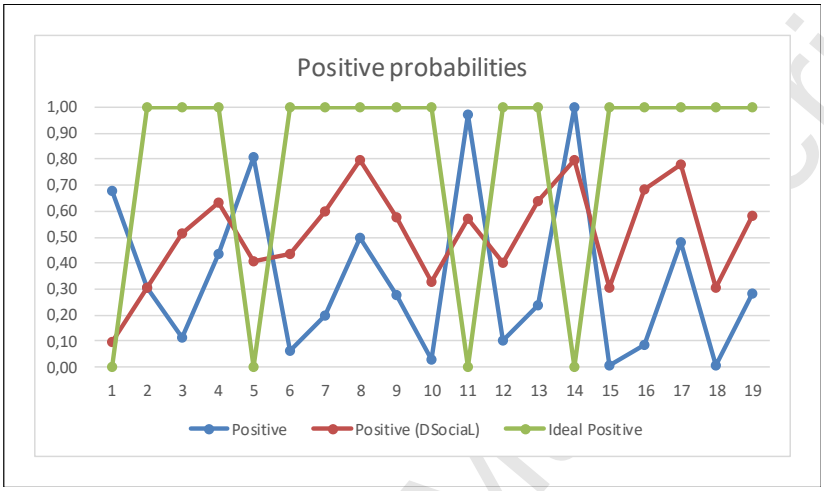


Figure 4: Comparison of positive probabilities

## 5. Conclusions and Future Work

Opinion mining is key to recommendation or feedback systems, in which the sentiment towards a product or service can be decisive in future actions of customers and providers. Classification methods are used to perform opinion mining to identify and extract subjective information in different sources. Moreover, probabilistic classifiers are reliable, but depend heavily on the size of the training set and the context in which they are applied, not fitting well with real time data. That is the reason why preprocessing is very important to improve the performance of classifiers and why there are a great number of research works that focus on preprocessing and related tasks. However, it should be done before training the classifier, which forces to retrain the classifier when any improvement in the preprocessing task is made. That is a very expensive task both in terms of time and in terms of computational cost.

Besides, postprocessing can improve the accuracy of classifiers, applying expert domain-specific knowledge about a new event or product. Thus, in this work, we propose a novel approach to improve the accuracy of decision support systems for sentiment analysis. We introduce the DSociaL platform, focusing on the parts of the system that are intended to improve the performance of sentiment analysis classifiers.

Our approach allows experts in specific domains of knowledge to identify positive and negative features that current classifiers are not able to automatically identify to correctly classify the polarity of a given text. By using a DSL, even people without a background

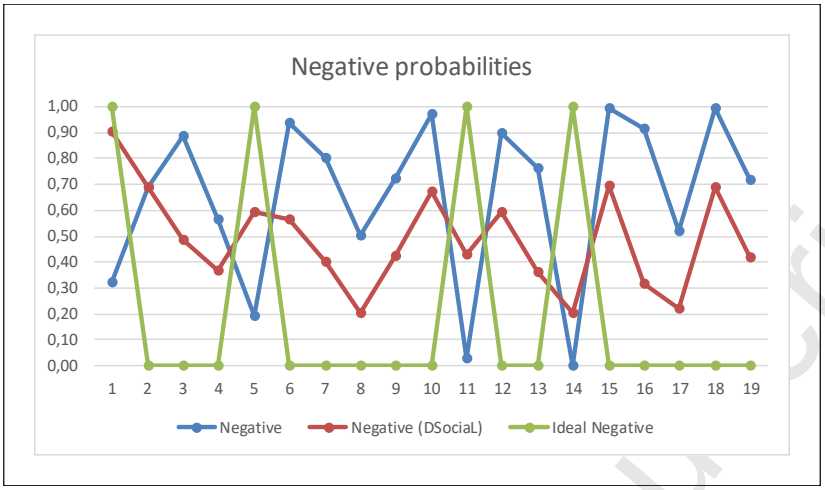


Figure 5: Comparison of negative probabilities

in traditional programming languages can use it to define rules and metarules that can improve the performance of a given classifier.

After training different messages from users on Twitter as positive or negative towards different movies, we show an evaluation to compare the results of the classifier to the results using DSociaL and expert knowledge. To that end, we use the cosine similarity between the analyzed messages and a set of text snippets and logical expressions. This set could be, for example, few phrases or expressions related to a current event. The output modifies directly the probability for a message to be positive, negative o neutral. So, this solution can be used after the machine learning algorithm for detecting the message polarization.

Future work will focus on different aspects. For example, we will work on different algorithms to match the similarity of two sentences or a fragment of text in a sentence. That way, the similarity percentage will behave better for the different definitions of the experts. We will also work with different classifiers and different configurations to try to determine in what factor, the knowledge of an expert can be decisive in a strongly trained classifier. We will also work on different components, such as the DSL to extract data from social networks, focusing not only on Twitter, but on other social networks, trying to establish relationships between the different data offered by them.

## References

[1] E. C. Wenger, W. M. Snyder, Communities of practice: The organizational frontier, Harvard business review 78 (1) (2000) 139–146.

[2] N. B. Ellison, B. D. M., Social network sites: Definition, history, and scholarship, Journal of Computer-Mediated Communication 13 (1) (2007) 210–230.

[3] C. M. Cheung, P.-Y. Chiu, M. K. Lee, Online social networks: Why do students use facebook?, Computers in Human Behavior 27 (4) (2011) 1337–1343.

[4] J. Bernabé-Moreno, A. Tejeda-Lorente, C. Porcel, H. Fujita, E. Herrera-Viedma, Caresome: A system to enrich marketing customers acquisition and retention cam-

paigns using social media information, Knowledge-Based Systems 80 (2015) 163–179. doi:http://dx.doi.org/10.1016/j.knosys.2014.12.033.

[5] J. Bernabé-Moreno, A. Tejeda-Lorente, C. Porcel, E. Herrera-Viedma, A new model to quantify the impact of a topic in a location over time with social media, Expert Systems with Applications 42 (7) (2015) 3381 – 3395. doi:http://dx.doi.org/10.1016/j.eswa.2014.11.067.

[6] D. Bell, T. Koulouri, S. Lauria, R. D. Macredie, J. Sutton, Microblogging as a mechanism for humanrobot interaction, Knowledge-Based Systems 69 (0) (2014) 64 – 77. doi:http://dx.doi.org/10.1016/j.knosys.2014.05.009.

[7] H. Cordobés, A. F. Anta, L. F. Chiroque, F. P. García, T. Redondo, A. Santos, Graph-based techniques for topic classification of tweets in spanish, IJIMAI 2 (5) (2014) 32–38. doi:http://dx.doi.org/10.9781/ijimai.2014.254.

[8] A. Java, X. Song, T. Finin, B. Tseng, Why we twitter: understanding microblogging usage and communities, in: Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 workshop on Web mining and social network analysis, ACM, 2007, pp. 56–65.

[9] H. Kwak, C. Lee, H. Park, S. Moon, What is twitter, a social network or a news media?, in: Proceedings of the 19th international conference on World wide web, ACM, 2010, pp. 591–600.

[10] M. E. Newman, J. Park, Why social networks are different from other types of networks, Physical Review E 68 (3) (2003) 036122.

[11] J. M. Kleinberg, Challenges in mining social network data: processes, privacy, and paradoxes, in: Proceeding KDD '07 Proceedings of the 13th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, 2007, pp. 4–5.

[12] I. H. Witten, E. Frank, M. A. Hall, Data Mining: Practical Machine Learning Tools and Techniques, 3rd Edition, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2011.

[13] U. Fayyad, G. Piatetsky-Shapiro, P. Smyth, The kdd process for extracting useful knowledge from volumes of data, Communications of the ACM 39 (11) (1996) 27–34.

[14] M. Kantardzic, Data mining: concepts, models, methods, and algorithms, John Wiley & Sons, 2011.

[15] R. Dale, H. Moisl, H. Somers, Handbook of natural language processing, CRC Press, 2000.

[16] B. Pang, L. Lee, Opinion mining and sentiment analysis, Foundations and trends in information retrieval 2 (1-2) (2008) 1–135.

[17] A. L. Samuel, Some studies in machine learning using the game of checkers, IBM Journal of research and development 3 (3) (1959) 210–229.

[18] V. Jagtap, K. Pawar, Analysis of different approaches to sentence-level sentiment classification, International Journal of Scientific Engineering and Technology 2 (3) (2013) 164–170.

[19] L. Walton, Domain-specific design languages, http://www.cse.ogi.edu/walton/dsdls.html.

[20] J. Bentley, Programming pearls: little languages, Communications of the ACM 29 (8) (1986) 711–721.

[21] A. van Deursen, P. Klint, Little languages: Little maintenance?, Journal of software maintenance 10 (2) (1998) 75–92.

[22] V. García-Díaz, H. Fernández-Fernández, E. Palacios-González, B. C. P. G-Bustelo, O. Sanjuan-Martínez, J. M. C. Lovelle, Talisman mde. mixing mde principles, Journal of Systems and Software 83 (7) (2010) 1179–1191.

[23] V. García-Díaz, J. B. Tolosa, B. C. P. G-Bustelo, E. Palacios-González, Ó. Sanjuan-Martínez, R. G. Crespo, Talisman mde framework: An architecture for intelligent model-driven engineering, in: International Work-Conference on Artificial Neural Networks, Springer Berlin Heidelberg, 2009, pp. 299–306.

[24] J. A. Balazs, J. D. Velásquez, Opinion mining and information fusion: a survey, Information Fusion 27 (2016) 95–110.

[25] A. Montejo-Ráez, E. Martínez-Cámara, M. T. Martín-Valdivia, L. A. Ureña-López, Ranked wordnet graph for sentiment polarity classification in twitter, Computer Speech & Language 28 (1) (2014) 93–107.

[26] K. Ravi, V. Ravi, A survey on opinion mining and sentiment analysis: tasks, approaches and applications, Knowledge-Based Systems 89 (2015) 14–46.

[27] B. Liu, L. Zhang, A survey of opinion mining and sentiment analysis, in: Mining text data, Springer, 2012, pp. 415–463.

[28] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, M. Stede, Lexicon-based methods for sentiment analysis, Computational linguistics 37 (2) (2011) 267–307.

[29] D. Vilares, C. Gómez-Rodríguez, M. A. Alonso, Universal, unsupervised (rule-based), uncovered

sentiment analysis, Knowledge-Based Systems 118 (2017) 45–55.
[30] S. Zhou, Q. Chen, X. Wang, Fuzzy deep belief networks for semi-supervised sentiment classification, Neurocomputing 131 (2014) 312–322.
[31] C. Catal, M. Nangir, A sentiment classification model based on multiple classifiers, Applied Soft Computing 50 (2017) 135–141.
[32] S. Alonso, I. J. Pérez, F. J. Cabrerizo, E. Herrera-Viedma, A linguistic consensus model for web 2.0 communities, Applied Soft Computing 13 (1) (2013) 149–157.
[33] N. Godbole, M. Srinivasaiah, S. Skiena, Large-scale sentiment analysis for news and blogs., ICWSM 7 (21) (2007) 219–222.
[34] C. B. González, J. García-Nieto, I. N. Delgado, J. F. A. Montes, A fine grain sentiment analysis with semantics in tweets, IJIMAI 3 (6) (2016) 22–28. doi:http://dx.doi.org/10.9781/ijimai.2016.363.
[35] B. Pang, L. Lee, S. Vaithyanathan, Thumbs up?: sentiment classification using machine learning techniques, in: Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10, Association for Computational Linguistics, 2002, pp. 79–86.
[36] A. Kennedy, D. Inkpen, Sentiment classification of movie reviews using contextual valence shifters, Computational intelligence 22 (2) (2006) 110–125.
[37] A. Pak, P. Paroubek, Twitter as a corpus for sentiment analysis and opinion mining., in: LREc, Vol. 10, 2010, pp. 1320–1326.
[38] S. Matsumoto, H. Takamura, M. Okumura, Sentiment classification using word sub-sequences and dependency sub-trees, in: Pacific-Asia Conference on Knowledge Discovery and Data Mining, Springer, 2005, pp. 301–311.
[39] R. Xia, C. Zong, S. Li, Ensemble of feature sets and classification algorithms for sentiment classification, Information Sciences 181 (6) (2011) 1138–1152.
[40] A. Abbasi, H. Chen, A. Salem, Sentiment analysis in multiple languages: Feature selection for opinion classification in web forums, ACM Transactions on Information Systems (TOIS) 26 (3) (2008) 12.
[41] H. Saif, Y. He, M. Fernandez, H. Alani, Contextual semantics for sentiment analysis of twitter, Information Processing & Management 52 (1) (2016) 5–19.
[42] H. G. Yoon, H. Kim, C. O. Kim, M. Song, Opinion polarity detection in twitter data combining shrinkage regression and topic modeling, Journal of Informetrics 10 (2) (2016) 634–644.
[43] D. Bollegala, D. Weir, J. Carroll, Cross-domain sentiment classification using a sentiment sensitive thesaurus, IEEE transactions on knowledge and data engineering 25 (8) (2013) 1719–1731.
[44] A. Balahur, G. Jacquet, Sentiment analysis meets social media challenges and solutions of the field in view of the current information sharing context, Information Processing & Management 51 (4) (2015) 428 – 432. doi:http://dx.doi.org/10.1016/j.ipm.2015.05.005.
[45] E. Haddi, X. Liu, Y. Shi, The role of text pre-processing in sentiment analysis, Procedia Computer Science 17 (2013) 26–32.
[46] T. Singh, M. Kumari, Role of text pre-processing in twitter sentiment analysis, Procedia Computer Science 89 (2016) 549–554.
[47] C. Lifna, M. Vijayalakshmi, Identifying concept-drift in twitter streams, Procedia Computer Science 45 (2015) 86–94.
[48] Y. Bao, C. Quan, L. Wang, F. Ren, The role of pre-processing in twitter sentiment analysis, in: International Conference on Intelligent Computing, Springer, 2014, pp. 615–624.
[49] G. Angiani, L. Ferrari, T. Fontanini, P. Fornacciari, E. Iotti, F. Magliani, S. Manicardi, A comparison between preprocessing techniques for sentiment analysis in twitter, in: Kdweb 2016.
[50] T. Chen, R. Xu, Y. He, X. Wang, Improving sentiment analysis via sentence type classification using bilstm-crf and cnn, Expert Systems with Applications 72 (2017) 221–230.
[51] P. Hudak, Domain-specific languages, Handbook of Programming Languages 3 (1997) 39–60.
[52] A. van Deursen, P. Klint, J. Visser, Domain-specific languages: an annotated bibliography, SIGPLAN Not. 35 (6) (2000) 26–36. doi:http://doi.acm.org/10.1145/352029.352035.
[53] M. Mernik, J. Heering, A. M. Sloane, When and how to develop domain-specific languages, ACM Comput. Surv. 37 (4) (2005) 316–344. doi:http://doi.acm.org/10.1145/1118890.1118892.
[54] B. A. Nardi, A small matter of programming: perspectives on end user computing, MIT press, 1993.
[55] V. García-Díaz, J. P. Espada, B. C. P. G. Bustelo, J. M. C. Lovelle, Towards a standard-based domain-specific platform to solve machine learning-based problems, IJIMAI 3 (5) (2015) 6–12. doi:http://dx.doi.org/10.9781/ijimai.2015.351.
[56] S. Efftinge, M. Völter, oaw xtext: A framework for textual dsls, in: Workshop on Modeling Symposium at Eclipse Summit, Vol. 32, 2006, p. 118.

[57] A. Van Deursen, P. Klint, Domain-specific language design requires feature descriptions, CIT. Journal of computing and information technology 10 (1) (2002) 1–17.
[58] A. Go, R. Bhayani, L. Huang, Twitter sentiment classification using distant supervision, CS224N Project Report, Stanford 1 (2009) 12.
[59] E. Kouloumpis, T. Wilson, J. D. Moore, Twitter sentiment analysis: The good the bad and the omg!, Icwsm 11 (2011) 538–541.
[60] M. Yassine, H. Hajj, A framework for emotion mining from text in online social networks, in: 2010 IEEE International Conference on Data Mining Workshops, IEEE, 2010, pp. 1136–1142.
[61] M. Damashek, Gauging similarity with n-grams: Language-independent categorization of text, Science 267 (5199) (1995) 843.
[62] S. Bird, Nltk: the natural language toolkit, in: Proceedings of the COLING/ACL on Interactive presentation sessions, Association for Computational Linguistics, 2006, pp. 69–72.
[63] L. Richardson, S. Ruby, RESTful web services, " O'Reilly Media, Inc.", 2008.
[64] M. Stonebraker, Sql databases v. nosql databases, Communications of the ACM 53 (4) (2010) 10–11.
[65] B. Amos, H. Turner, J. White, Applying machine learning classifiers to dynamic android malware detection at scale, in: 2013 9th international wireless communications and mobile computing conference (IWCMC), IEEE, 2013, pp. 1666–1671.
[66] G. P. Petropoulos, K. Arvanitis, N. Sigrimis, Hyperion hyperspectral imagery analysis combined with machine learning classifiers for land use/cover mapping, Expert systems with Applications 39 (3) (2012) 3800–3809.
[67] R. Ramos-Pollán, M. A. Guevara-López, C. Suárez-Ortega, G. Díaz-Herrero, J. M. Franco-Valiente, M. Rubio-del Solar, N. González-de Posada, M. A. P. Vaz, J. Loureiro, I. Ramos, Discovering mammography-based machine learning classifiers for breast cancer diagnosis, Journal of medical systems 36 (4) (2012) 2259–2269.
[68] G. Ateniese, L. V. Mancini, A. Spognardi, A. Villani, D. Vitali, G. Felici, Hacking smart machines with smarter ones: How to extract meaningful data from machine learning classifiers, International Journal of Security and Networks 10 (3) (2015) 137–150.
[69] I. Rish, An empirical study of the naive bayes classifier, in: IJCAI 2001 workshop on empirical methods in artificial intelligence, Vol. 3, IBM New York, 2001, pp. 41–46.
[70] S. A. Pattekari, A. Parveen, Prediction system for heart disease using naïve bayes, International Journal of Advanced Computer and Mathematical Sciences 3 (3) (2012) 290–294.
[71] B. Liu, E. Blasch, Y. Chen, D. Shen, G. Chen, Scalable sentiment classification for big data analysis using naive bayes classifier, in: Big Data, 2013 IEEE International Conference on, IEEE, 2013, pp. 99–104.
[72] V. Narayanan, I. Arora, A. Bhatia, Fast and accurate sentiment classification using an enhanced naive bayes model, in: International Conference on Intelligent Data Engineering and Automated Learning, Springer, 2013, pp. 194–201.
[73] L. Jiang, Z. Cai, H. Zhang, D. Wang, Naive bayes text classifiers: a locally weighted learning approach, Journal of Experimental & Theoretical Artificial Intelligence 25 (2) (2013) 273–286.