International Journal of
Computer & Software Engineering

Research Article Open Access

# Can Opinion Mining Techniques Help to Select Open Source Software?

**Davide Taibi**
*Faculty of Computer Science, Free University of Bozen, 1, Bolzano BZ, Italy*

## Abstract

People and organizations that are considering the adoption of Open-Source Software (OSS), or that need to choose among different OSS products are interested in knowing the user community's opinion, since this can provide useful indications about the strengths and limits of the software being evaluated. While several methods for the evaluation of the community size are available, there is no automated support to the extraction of the opinions of the community. In this paper we explore whether it is possible to support the OSS selection process by means of automated sentiment analysis techniques. Our goal is to understand if the actual opinion mining techniques, can be applied to get valuable opinions on OSS software. Our goal will be achieved first developing a web crawler to extract user generated content on OSS, building a data-set of relevant user generated then we apply the opinion mining process the OSS blogs data-set. We collected more than 88K user generated content and we compared the performance of our opinion mining technique with a set of existing opinion mining tools. Results of the application of our technique show that opinion mining can help to evaluate the opinions of OSS products. However, the existing opinion mining tools, even if applicable in different domains, are still not reliable in the domain of OSS, mainly because they are trained on different data-sets, opening new research directions for future work in the opinion mining domain.

**Publication History:**

Received: January 20, 2016
Accepted: June 08, 2016
Published: June 10, 2016

**Keywords:**

Open Source Adoption, Open Source Selection, Opinion Mining, Sentiment Analysis

## Introduction

With the born of several blogging platforms (see for instance Wordpress1) users express their opinions on every kind of topics, from politic to religion, from marketing to product reviews, etc. This wealth of opinionated contents can be very useful to catch the user's thought and perception. We are interested in analyzing opinions of OSS because of the nature of its selection. The software selection process, and particularly OSS, often includes a preliminary phase in which the potential users collect information about the products by surfing in blogs, forums and newsgroups. Thus, the process of selecting OSS very often involves a long and boring web search, looking into several forums, blogs and websites in order to extract as much information as possible.

For this reasons, we aim at understanding if we can apply opinion mining techniques, so as to support the OSS adoption process by understanding the general opinions on a given OSS.

The paper is structured as follow. In Section 2 we present background and related works. In Section 3 we summarize the approach used in this work. Section 4 we present a case study comparing our opinion mining approach with a set of opinion mining tools while in Section 5 discuss the results. Finally, in Section 6, we draw conclusions and we outline future works.

## Background and Related Work

### Open source adoption and adoption motivations

The definition of the information commonly used by the users when they evaluate OSS projects has been investigated in the last few years, and several OSS evaluation methods have been proposed [1-3]. The reasons and motivations that lead software companies to adopt or reject OSS and to understand how people can trust software. Moreover, several empirical works investigated the importance of OSS and the factors that bring to the OSS adoption.

In our previous work [4,5] we conducted a survey with 151 FLOSS stakeholders, with different roles and responsibilities, about the factors that they consider most important when assessing whether FLOSS is trustworthy. Here, we did not ask the motivations for the adoption of a FLOSS product or a proprietary one, but we asked for the factors considered to compare two FLOSS products.

We identified 37 factors, clustered in five groups: economic, license, development process, product quality, customer-oriented requirements and user opinions was considered of middle importance. Moreover, in another recent work, [6] we ran another survey to identify the recent motivations for the adoption of OSS and also in this case, the opinions were also considered as important. Results are also confirmed by several empirical study such as [7-13]. The idea of using opinion mining techniques have been proposed in several work [13,14] Based on the aforementioned work, we believe that the investigation of an automated opinion extraction tool would be valuable for OSS users [6].

### Opinion mining and sentiment analysis approaches

Recently, a good deal of work has been done by researchers on sentiment analysis (or polarity analysis) of reviews and opinion mining. Opinion mining is an important step where researchers try to understand if a text contains an opinion while Sentiment Analysis is a step further, involving polarity analysis detecting if an opinion is positive or negative.

Typically, the methods employed include combinations of machine learning and shallow natural language processing methods, and achieve good accuracy [15]. For instance, a study showed that peaks in references to books in weblogs are likely to be followed by peaks in their sales [16].

The year 2001 or so, seems to be the beginning of widespread awareness of the research problems and opportunities that sentiment analysis and opinion mining raise. Factors behind this widespread include the rise of machine learning methods in natural language

*Corresponding Author: Dr. Davide Taibi, Faculty of Computer Science, Free University of Bozen, 1, Bolzano BZ, Italy; E-mail: davide.taibi@unibz.it

**Citation:** Taibi D (2016) Can Opinion Mining Techniques Help to Select Open Source Software?. Int J Comput Softw Eng 1: 104. doi: http://dx.doi.org/10.15344/ijcse/2016/104

processing and information retrieval, the availability of dataset for machine learning algorithms to be trained. In 2002, Bo Pang and Lillian Lee [17], applied three Machine Learning techniques on the Movie review Domain:

Support Vector Machines (SVMs), Naive Bayes and Maximum Entropy. They tested the algorithms on unigrams and bigrams, appending POS tags to every word via Oliver Mason's QTag program [18]. This serves a crude form of word sense disambiguation distinguishing the different usage of words (e.g. the difference in usages of the word "love" in "I love this movie" versus "This is a love story". They looked at the performance of using adjectives alone, discovering that adjectives provided a less useful information then unigram presence. Indeed, simply using the most frequent unigrams presence information turned out to be the most effective way to spot opinions in text, yielding performance comparable to that using the presence of all lines. In terms of relative performance, Naive Bayes tend to the worst while Support Vector Machines tend to the best. An important problem come out from this paper was the needs of the identification of some kind of features indicating weather sentences are on topic.

Since 1992, the born of challenges (see for instance TREC2 and SIGIR3) encourage research within the information retrieval community by providing the infrastructure necessary for large-scale evaluation of text retrieval methodologies. For Example, each year, TREC provides a test set of documents and questions on which participants can run their own retrieval systems, and return a list of the retrieved top-ranked documents. Then, TREC collect and judges the retrieved documents for correctness, and evaluates the results. The TREC cycle ends with a workshop that is a forum for participants to share their experiences. TREC and other similar contexts help significantly the community in improve their Information Retrieval algorithms by means of the comparison of the results of several people on the same problem. The vast majority of Opinion Mining techniques applied a 3-step algorithm. In [19] Zhang et al, first decompose documents in sentences, then labeled each sentence with an opinion score by means of a classifier. Sequently they labeled the text if it contains at least one opinionated sentence. Yang et al in [20] also considered multiple sources of evidence in their opinion retrieval steps by combining scores from opinion detection based on a common opinions terms. Then they built a lexicon by identifying the most occurring terms. Sequently, the opinion terms are manually labeled by assigning an opinion weight. They identify also the low frequency terms score, and acronyms score. Moreover, they combined the obtained each score as a weighted sum and they used this sum as training data. Finally in the third step, the linear combination of relevance and opinion score is used to score and rank documents. In another paper, Zhang et al. [21] used the Classification by Minimizing Errors (CME) to assign an opinion to each sentence of a blog post. They assigned an opinion score to each document by means of an SVN classifier, based on the values of the defined features. They used a set of movie review4 to train the CME classifier and they used a labelled dataset for classifying documents by means of SVM. The blog posts were ranked by the final score that was calculated as the relevance score times the opinion score.

Our work differs from existing studies of sentiment analysis and business data in two important aspects. First, our domain includes weblogs from several sources i.e., a set of domains which tend to be far less focused and organized than the typical product review data targeted by sentiment analyzers, and consist predominantly of informal and unorganized text. Second, we aim at applying this study as a specific means to understand if the opinion of the OSS community on an OSS product reflects the software quality.

## The Study Approach

In this section we summarize the approach adopted for this case study. We first extract the relevant blog post from existing search engines, then we run the distillation task, to retrieve only the "opinionated" content, and finally we apply the opinion retrieval technique.

### Web crawling

In order to build data-set containing posts that express an opinion about a particular topic, we developed a web crawler that queries blog search engines (like Technorati 5 and Google log Search 6) every day, extracting the list of relevant posts for a given topic.

Each data-set includes an xml index summarizing each post, a folder containing all html pages, and a database dump with some meta-information. The crawler stores all post information on a relational database to speed up the post selection up by means of SQL queries. Furthermore, the crawler extracts pages by using the same format of TREC, so the application of techniques developed in TREC can be quickly applied to the generated data-sets.

### Blog distillation

Blog search users often wish to identify blogs about a given topic so that they can subscribe to them and read them on a regular basis [22]. The blog distillation task can be defined as: "Find me a blog with a principle, recurring interest in the topic X." Systems should suggest feeds that are principally devoted to the topic over the time span of the feed, and would be recommended to a user as an interesting feed about the topic (i.e. a user may be interested in adding it to his RSS reader).

The blog distillation task has been approached from many different points of view. In [23], the authors view the distillation task as an as ad-hoc search and they consider each blog as a long document created by concatenating all postings together. Other researchers treat it as the resource ranking problem in federated search [24]. They view the blog search problem as the task of ranking collections of blog posts rather than single documents. A similar approach has been used in [25], where they consider a blog as a collection of postings and use resource selection approaches. Their intuition is that finding relevant blogs is similar to finding relevant collections in a distributed search environment. In [22], the authors modeled blog distillation as an expert search problem and use a voting model for tackling it.

Our intuition is that each post in a blog provides evidence regarding the relevancy of that blog to a specific topic. Blogs with more (positive) evidence are more likely to be relevant. Moreover, each post has many different features like content, in links, and anchor text that can be used to estimate relevancy. There are also global features of each blog like the total number of posts, the number of postings that are relevant to the topic and the cohesiveness of the blog that could be useful to consider.

Our first approach is to create a baseline for blog distillation system that uses only the content of blog posts as a source of evidence. To do this, we consider the expert search idea proposed in [26]. The main idea of that work is to treat blogs as experts and feed distillation as expert search. In the expert search task, systems are asked to rank candidate experts with respect to their predicted expertise about a query, using documentary evidence of expertise found in the collection. So the idea is that the blog distillation task can be seen as a voting process: A blogger with an interest in a topic would send a post regularly about the topic, and these blog posts would be retrieved in response to the query. Each time a blog post is retrieved, it can be seen as a vote for that blog as being relevant to (an expert in) the topic area. We use the voting model to find relevant blogs. The model ranks blogs by considering the sum of the exponential of the relevance scores of the postings associated with each blog. The model is one of the data fusion models which Macdonald and Ounis used in their expert search system [27]. For our second approach, we used more features to represent each blog beside its content. To take the different features into account, we use a Rank Learning approach [28] to combine the features into a single retrieval function. Useful features are:

- Cohesiveness of blog postings
- Number of postings
- Number of relevant postings (posts in top N relevance results)
- Number of in links
- Relevance of in link post content
- Relevance of in link anchor-text

## Opinion mining

In the Opinion Retrieval task, we adopt the approach defined in [29]. Here, we report on the main steps of the approach. More details can be found in [29]. The approach is composed by 2 steps: First we index the collection, then we combine additional information, including the content of incoming hyperlinks and tag data from social bookmarking websites with our basic retrieval method (Divergence from Randomness version of BM25 (DFR BM25) (table 1) [17].

| Run | Mean Average Precision | Recall Precision | Recall Precision @10 |
|---|---|---|---|
| DFR BM25<br>our baseline system | 0.2138<br>0.2663 | 0.3836<br>0.2780 | 0.4087<br>0.4273 |

Table 1: Topic-Relevance results for submitted baseline.

The collection indexing phase is carried out by means of Terrier Information Retrieval system [30]. We extended this content-based retrieval technique with additional information including the content of incoming hyperlinks and tag data from social bookmarking websites. The latter has been shown to be useful for improving Web Search [26, 31-33].

Our approach to ranking blog posts by their opinion level relies on a learning framework [28,34]. We trained a Learning to Rank system to take both a relevance score (output by the rank learner described above) and an ”opinion score” for each document into account when producing an output ranking. The advantage of this approach is that we do not need to explicitly decide how to combine these forms of evidence, but can rely on historical data for fine tuning the retrieval.

The problem is to estimate a score for the”opinion atedness” of each document. We have two approaches to doing this. In the first approach, we calculate an opinion score for each term in the document and then we combine the score over all terms in the document. In the second, we train a classification system to distinguish between opinionated and non-opinionated posts. Then, we use the confidence of the classifier as an opinion score for the document.

The opinion score is then calculated with two methods. The first method considers the technique proposed by Amati [35] and the Kullback-Leibler divergence [36] between the opinionated document set and the relevant document set as:

$$Opinion1(d) = \sum_{t \in d} opinion(t)\, p(t/d)$$

where p(t|d) is the relative frequency of term t in the document d.

As second opinion retrieval method, we trained a Support Vector Machine (SVM) to recognize opinionated documents. We can then use the confidence of the classifier (i.e. the distance from the hyperplane) as the opinion score for each document. The per term opinion score is used in this case only for feature selection, As second opinion retrieval method, we trained a Support Vector Machine (SVM) to recognize opinionated documents. We can then use the confidence of the classifier (i.e. the distance from the hyperplane) as the opinion score for each document. The per term opinion score is used in this case only for feature selection,

$$Opinion\_SVM(d) = f_{SVM}(p(t_1/d), \ldots, p(t_m/d))$$

## Results

In this section, we describe the results of the application of our approach to a set of OSS projects with our opinion mining techniques (Opin1 and OpinSVM) and four of the most common opinion mining tools:

- Python NLTK Text Classification[5]
- Vivekl[6]
- SentiStrenght[7]
- SentiRank[8]

The goal is to understand the power of our techniques, compared with existing ones, so as to understand if using a generic tool, not trained for our purpose, would provide similar results.

| First Word | Second Word |
|---|---|
| JJ | NN or NNS |
| RB, RBR, or RBS | JJ |
| JJ | JJ |
| NN or NNS | JJ |
| RB, RBR or RBS | VBN or VBG |

Table 2: Pattern of tags for extracting two-word phrases.

## OSS data-set building

The first step to be carried out is the selection of project and the online extraction of user generated content. The selection of projects addressed different types of software applications, generally considered stable and mature. The complete set of projects comprises 27 products, having different characteristics:

There are different types of programs and applications (from web servers to operating system, from libraries to content management systems).

The communities of developers and users have different sizes. – The projects have different ages.

During the selection it was taken care that every factor value was present in the set of projects. The crawler ran for 4 months, collecting a total of 78531 posts with thousands of posts for each OSS project. In Table 3 we show the projects list and the number of posts extracted together with the average sentiment obtained by each approach adopted in Section 5. In Figure 1 we show the trend of the number of posts retrieved in the same period.

In Figure 1 we can see an interesting trend for each project, considering the number of extracted post per day. What we can clearly see is that there are no major intersections and all projects follow a similar trend. We reserve the possibility of more investigation in future works.

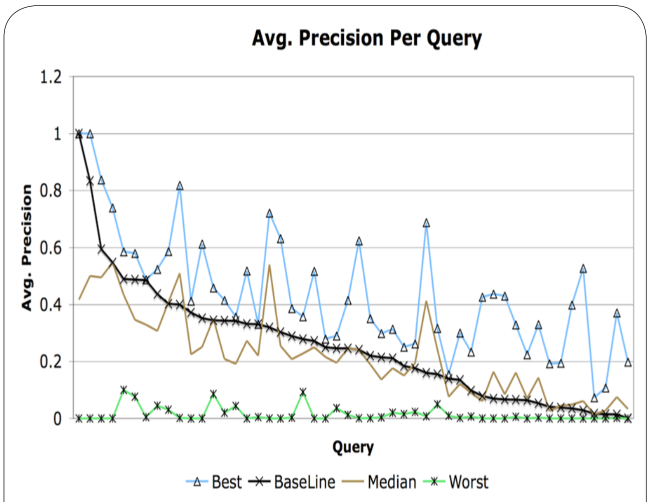

Figure 1: Average Precision for each Query (ordered by relative performance) for the blog distillation task using a simple voting model. The best, median and worst scores are those of the other participants in TREC'08.

### Opinion mining

In order to evaluate the results, we manually labelled 2700 posts (100 per project) Here we first apply the tools and techniques to the whole data-set, in order to compare the resulting polarity of each post. Then we apply again the tools on a manually labelled analyze reported Results for the polarity obtained by means of the application of the Python NLTK tool to our complete data-set are presented in Table 3 while results for precision and recall for the labelled data-set are reported in Table 4.

### Opin1 and OpinSVN

Here we report on the application of our first approaches as reported in Section 3. In order to calculate the opinion score for documents we used the expected opinionatedness of words in documents as described in Section 3. In OpinSVM we also used the confidence of the trained SVM to find the opinion score of the documents. We used the classifier to classify test document. The confidence of the classifier was then used as opinion score for documents.

Finally, we combine relevance and opinion score (from step 4) by means of SVM and we produce the final ranking. The collection indexing was carried out by means of Terrier Information Retrieval system [30]. Like in Section 2, we used the Divergence from Randomness version of BM25 (DFR BM25) weighting model to compute a score for each blog post.

The average precision of the Opin1 approach is 73% while the recall is 63% while for the OpinSVM approach we obtained a precision of 76% and a recall of 72%.

Finally, we tested the models (Opinion1 and Opinion SVM) by using the TREC'08 training data, so as to evaluate the capability of the algorithm of classifying opinions with a system trained on a different domain. As expected, in this case, the result show of a very low precision and recall (precision=0.66, recall=0.53). The reason is possibly due to the data-set itself. The TREC'08 data-set is composed by a heterogeneous set of topics, ranging from politics to science, from news to art. Our data-set is domain specific and need an ad-hoc training set to be set-up. We also think that a disadvantage of our"multiple-levels of learning" approach is that we cannot maximize the use of training data because of its unavailability.

### Python NLTK

The Python NLTK (NLTK) Text Classification applies an over 50 corpora and 10 lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum. It provides positive and negative polarity results that ranges from 0 to 1. The application to our trained data-set report a precision of 43% and a recall of 17%.

### Vivekn

Vivekn is based on a Naïve Bayes classifier and it examines individual words and short sequences of words (n-grams) and comparing them with a probability model. The probability model is built on a labelled data-set of IMDb movie reviews. It provides results with 3 possible labels, positive, negative or neutral, together with a confidence interval. The application to our trained data-set report a precision of 40% and a recall of 20%.

### SentiStrength

SentiStrength [37] estimates the strength of positive and negative sentiment in short texts, even for informal language. It claims to have a human-level accuracy for short social web texts in English, except for political texts.

SentiStrength reports values that range from -5 (extremely negative) to +5 (extremely positive). The application to our trained data-set report a precision of 45% and a recall of 24%.

### SentiRank

SentiRank uses a proprietary algorithm, which uses unspecified linguistic rules todetermine the sentiment. Results are presented with different ranges where values from -10 to 5 represent very negative results, values from -4.99 to -1 are related to negative opinions, values from -0.99 to +0.99 to neutral opinions, 1 to 4.99 for positive and finally from 5 to 10 for very positive opinions. The application to our trained data-set report a precision of 55% and a recall of 35%.

## Results Comparison

The selected approached and tools apply different algorithms, based on different trained data-sets. In order to compare the results, we analyze all post in our labeled data-set for each project with all tools and we normalized results with a scale that ranges from -1 (very negative) to +1 (very positive). In case of Vivekn, we assigned -1 to negative content, 0 to neutral and +1 but we also report the confidence level.

As expected, the results obtained from out techniques always identify the polarity more accurately, compared to the other tools. We believe this is due to the training on a domain-specific data-set (Table 3). Moreover, as we can see from Table 3 and Figure 2, results obtained from the four tools provide non homogeneous polarity results. Opin1 and OpinSVM always provide the same polarity while also NLTK provides a similar polarity of SentiStrenght. This again, can be due to the fact that the tools have been trained with a set of common datasets. Vivekn and SentiStrenght often provide divergent results from the other tools [38-40].

We believe that the application of our approach provides the better results compared to the other tools, mainly because it has been trained on a labelled dataset on the OSS domain.

Considering the application of our approach to the set of OSS projects, we find out that, at the moment, the opinion retrieval technique we adopted in TREC'08 [29] could be applicable to assess opinions on OSS project review, so as to help users to get a general overview on other users' opinions. Considering the other tools used in this study, we believe that, since they are claimed to be able to predict opinions with human-level accuracy, the reason of negative result is due to the fact that the actual opinion mining techniques has been designed for different domains and trained on different data-sets.

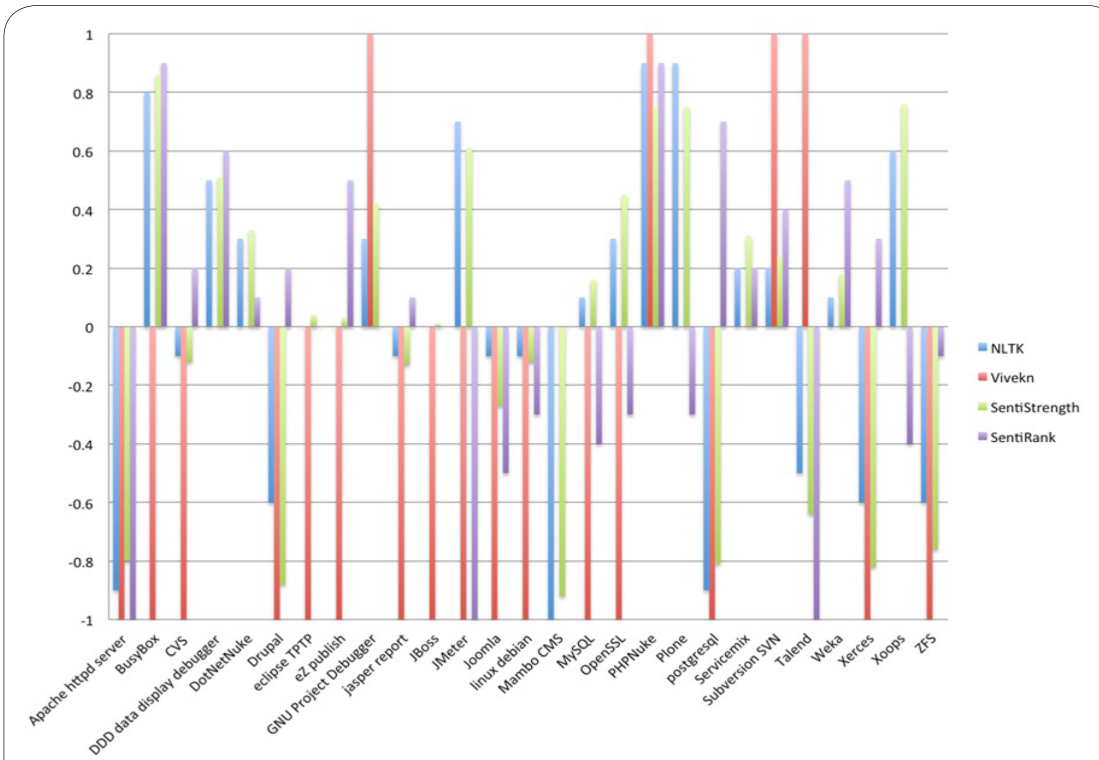

Figure 2: Sentiment Analysis with online tools comparison.

## Conclusion

In this work we investigated is is possible to use the actual opinion mining techniques to support the Open Source Software (OSS) adoption process. The goal is to understand if users can rely on opinions automatically collected from forums, so as to get an overall overview of the common opinion on a possible OSS they are interesting to adopt.

In this paper, we first developed a web crawler to extract user generated content online on 27 well known OSS projects from blogs and forum. We ran the crawler for four months, extracting 88K blog posts and then we evaluate the opinions of OSS users expressed in the documents. In order to evaluate the accuracy of the results, we manually labelled the relevance and the polarity of 2700 retrieved documents (100 per project). Then we compared the result obtained with four of the most common Opinion Mining tools with our proposed approaches [41-43].

Results show that our approach identifies the opinions with an acceptable accuracy (precision = 76% and recall=72%). The application of existing opinion mining tools provide very discordant results, only in few cases all the tools provide the same polarity for all the texts analyzed, reporting a very low precision and recall.

In order to rely on the Opinion Mining frameworks, our future work include the application of different opinion mining technique, and the development of an online tool to directly support users in the OSS evaluation.

## Acknowledgements

We are deeply indebted to and would like to thank University of Lugano, in quality of Prof. Fabio Crestani, Shima Gerani and Mark Carman for the collaboration in this work.

## Author Contributions

Conceived and designed the experiments: DT. Analyzed the data: DT. Wrote the first draft of the manuscript: DT. Contributed to the writing of the manuscript: DT. Agree with manuscript results and conclusions: DT. Jointly developed the structure and arguments for the paper: DT. Made critical revisions and approved final version: DT.

All authors reviewed and approved of the final manuscript.

## Disclosures and Ethics

As a requirement of publication author(s) have provided to the publisher signed confirmation of compliance with legal and ethical obligations including but not limited to the following: authorship and contributorship, conflicts of interest, privacy and confidentiality and (where applicable) protection of human and animal research subjects.

The authors have read and confirmed their agreement with the ICMJE authorship and conflict of interest criteria. The authors have also confirmed that this article is unique and not under consideration or published in any other publication, and that they have permission from rights holders to reproduce any copyrighted material. Any disclosures are made in this section. The external blind peer reviewers report no conflicts of interest.

## References

1. Taibi D, Lavazza Z, Morasca S (2007) OpenBQR: a framework for the assessment of oss. Int J Open Source Develop Adop Innov 173-186.
2. Lavazza L, Morasca S, Taibi D, Tosi D (2010) Applying SCRUM in an OSS Development Process: An Empirical Evaluation. 11th Int Con 48: 147-159.
3. Bianco VD, Lavazza L, Morasca S, Taibi, D (2009) Quality of Open Source Software: The QualiPSo Trustworthiness Model in OSS 199-212.
4. Bianco VD, Lavazza L, Morasca S, Taibi D (2011) A survey in Open Source Software Trustworthiness. IEEE Software 28: 5.
5. Bianco VD, Lavazza, L, Morasca S, Taibi D, Tosi D (2010) A Survey on the Importance of Some Economic Factors in the Adoption of Open Source Software. in SERA 151-162.
6. Taibi D (2015) An Empirical Investigation on the Motivations for the Adoption of Open Source Software. in ICSEA - The Tenth International Conference on Software Engineering Advances, Barcelona (Spain).
7. Tosi D, Lavazza L, Morasca S, Taibi D (2012) On the Definition of Dynamic Software Measures. in Proceedings of the ACM-IEEE International Symposium on Empirical Software Engineering and Measurement USA.
8. Taibi D, Lavazza L, Morasca S, Tosi D (2012) An Empirical Investigation of Perceived Reliability of Open Source Java Programs. in SAC '12 Proceedings of the 27th Annual ACM Symposium on Applied Comput Riva del Garda (Trento), Italy. 1109-1114
9. Lavazza L, Morasca S, Taibi D, Tosi D (2011) OP2A: How to Improve the Quality of the Web Portal of Open Source Software Products. in Lecture Notes in Business Information Processing, Springer-Verlag 101: 149-162.
10. Taibi D (2011) Towards a trust worthiness model for Open Source Software. LAP LAMBERT Academic Pub 232.
11. Lavazza L, Morasca S, Taibi D, Tosi D (2010) Predicting OSS trustworthiness on the basis of elementary code assessment.
12. Bianco VD, Lavazza L, Morasca S, Taibi D, Tosi D (2010) The QualiSPo approach to OSS product quality evaluation. Proceed 3rd Int Workshop on Emerg Trends, USA. OSS 23-28.
13. Taibi D, Bianco VD, Carbonare DD, Lavazza L, Morasca S (2008) Towards The Evaluation of OSS Trustworthiness: Lessons Learned From The Observation of Relevant OSS Projects 275: 389-395.
14. Bianco VD, Lavazza L, Morasca S, Taibi D, Tosi D (2010) The QualiSPo approach to OSS product quality evaluation. Proceedings of the 3rd International Workshop on Emerging Trends in Free/Libre/Open Source Software Research and Development, USA, 23-28.
15. Mishne G, Glance N (2006) Predicting movie sales from blogger sentiment. In AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs, 2006.
16. Gruhl D, Guha R, Kumar R, Novak J, Tomkins A (2005) The predictive power of online chatter. In KDD '05: Proceeding of the eleventh ACM SIGKDD international conference on Knowledge discovery in data mining, pages 78-87.
17. Amati G, Carpineto C, Romano G (2004) Fondazione ugo bordoni at trec 2004. In Proceedings of the Thirteenth Text REtrieval Conference (TREC 2004).
18. Mason O (1997) Qtag - a portable probabilistic tagger.
19. Zhang W, Clement Y, Meng W (2007) Opinion retrieval from blogs. In CIKM '07: Proceedings of the sixteenth ACM conference on Conference on information and knowledge management USA 831–840.
20. Yang K, Yu N, Zhang H (2007) Widit in trec 2007 blog track: Combining lexicon based methods to detect opinionated blogs. In Proceedings of the Sixteenth Text REtrieval Conference.
21. Zhang Q, Wang B, Wu L, Huang X (2007) Fdu at trec 2007: Opinion retrieval of blog track. In Proceedings of the Sixteenth Text REtrieval Conference.
22. Macdonald C, Ounis I, Soboroff I (2007) Overview of the trec-2007 blog track. In Proceedings of the Sixteenth Text REtrieval Conference (TREC 2007).
23. Efron M, Turnbull D, Ovalle C (2007) University of Texas School of Information at TREC 2007. In Proc of the 2007 Text REtrieval Conference.
24. Elsas J, Arguello J, Callan J, Carbonell J (2008) Re- trieval and feedback models for blog feed search. In SIGIR 347-354.
25. Seo J, Croft WB (2007) UMass at TREC 2007 Blog Distillation Task. In Proc. of the 2007 Text REtrieval Conference.
26. Hannah D, Macdonald C, Peng J, He B, Ounis I (2007) University of Glas-gow at TREC 2007: Experiments in Blog and Enterprise Tracks with Terrier. In Proceedings of TREC.
27. Macdonald C, Ounis I (2006) Voting for candidates: adapting data fusion techniques for an expert search task. In Proceedings of the 15th ACM international conference on Information and knowledge management USA 387-396.
28. Yue Y, Finley T, Radlinski F, Joachims T (2007) A support vector method for optimizing average precision. In SIGIR '07: Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval 271-278.
29. Gerani S, Keikha M, Carman M, Gwadera R, Taibi D, et al. (2008) University of lugano at trec 2008 blog track. In Ellen M. Voorhees and Lori P. Buckland, editors, TREC 500-277. National Institute of Stan- dards and Technology (NIST).
30. Ounis I, Amati G, Plachouras V, He B, Macdonald C, et al. (2006) Terrier: A High Performance and Scalable Information Retrieval Platform. In Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006).
31. Bao S, Xue G, Wu X, Yu Y, Fei B, et al. (2007) Optimizing web search using social annotations. In WWW '07: Proceedings of the 16th international conference on World Wide Web 501-510.
32. Heymann P, Koutrika G, Garcia-Molina H (2008) Can social book- marks improve web search? In WSDM '08: Proceedings of the First ACM Inter-national Conference on Web Search and Data Mining.
33. Yanbe Y, Jatowt A (2007) Satoshi Nakamura, and Katsumi Tanaka. Can social bookmarking enhance search in the web? In JCDL '07: Proceedings of the 2007 conference on Digital libraries 107–116.
34. Burges C, Shaked T, Renshaw E, Lazier A, Deeds M, et al. (2008) Learning to rank using gradient descent. In ICML '05: Proceedings of the 22nd international conference on Machine learning 89- 96.
35. Amati G, Ambrosi E, Bianchi M, Gaibisso C, Gambosi G (2008) Automatic construction of an opinion-term vocabulary for ad hoc retrieval. In Proceedings of the 30th European Conference on IR 4956: 89-100.
36. Manning CD, Schtze H (1999) Foundations of Statistical Natural Language Processing. The MIT Press.
37. Thelwall M, Buckley K, Paltoglou G, Cai D, Kappas A (2010) Sentiment strength detection in short informal text. J Am Soc Inform Sci Tech 61: 2544-2558.

38. Amati G, Rijsbergen CJV (2002) Probabilistic models of information retrieval based on measuring the divergence from randomness. ACM Transactions on Information Systems (TOIS) 20: 357-389.

39. Robertson S, Zaragoza H, Taylor M (2004) Simple bm25 extension to multiple weighted fields. In CIKM '04: Proceedings of the thirteenth ACM international conference on Information and knowledge management, pages 42-49.

40. Zhai C, Lafferty J (2004) A study of smoothing methods for language models applied to information retrieval. ACM Transactions on Information Sys- tems (TOIS) 22: 179-214.

41. Lenarduzzi V, Lunesu I, Matta M, Taibi D (2015) Functional Size Measures and Effort Estimation in Agile Development: a Replicated Study. in XP2015.

42. Bianco VD, Lavazza L, Lenarduzzi V, Morasca S, Taibi D, et al. (2012) A Study on OSS Marketing and Communication Strategies. in 8th IFIP International Conf Open Source Soft OSS Open Source Systems: Long-Term Sustainability 378: 338-343.

43. Bianco VD, Lavazza L, Morasca S, Taibi D, Tosi D (2010) An Investigation of the Users' Perception of OSS Quality. OSS 319: 15-28.

| OSS Project | #Retrieved Posts | Opin1 | OpinSVM | NLTK (sentiment) | Vivekl | | Senti Strength (sentiment) | Senti Rank (sentiment) |
|---|---|---|---|---|---|---|---|---|
| | | | | | sentiment | conf. | | |
| Apache httpd server | 3550 | -0.57 | -1.21 | -0.9 | -1 | 99% | -0.8 | -1.0 |
| BusyBox | 1169 | 0.99 | 0.28 | 0.8 | -1 | 99% | 0.86 | 0.9 |
| CVS | 3564 | 0.09 | 0.07 | -0.1 | -1 | 98% | -0.12 | 0.2 |
| DDD data display debugger | 3598 | 0.92 | -0.35 | 0.5 | 0 | 97% | 0.51 | 0.6 |
| DotNetNuke | 1866 | 0.64 | -0.22 | 0.3 | 0 | 92% | 0.33 | 0.1 |
| Drupal | 8904 | -0.65 | 0.14 | -0.6 | -1 | 98% | -0.88 | 0.2 |
| eclipse TPTP | 2695 | -0.16 | 0.21 | 0 | -1 | 99% | 0.04 | 0.0 |
| eZ publish | 2612 | 0.14 | 0.01 | 0 | -1 | 98% | 0.03 | 0.5 |
| GNU Project Debugger | 2784 | 0.09 | 0.34 | 0.3 | -1 | 98% | 0.42 | 0.0 |
| jasper report | 2674 | 0.15 | -0.12 | -0.1 | -1 | 97% | -0.13 | 0.1 |
| JBoss | 3557 | -0.23 | 0.02 | 0 | -1 | 99% | 0.01 | 0.0 |
| JMeter | 524 | 0.91 | 0.12 | 0.7 | -1 | 99% | 0.61 | -1.1 |
| Joomla | 9477 | -0.16 | -0.15 | -0.1 | -1 | 99% | -0.27 | -0.5 |
| linux debian | 2305 | -0.2 | 0.35 | -0.1 | -1 | 98% | -0.12 | -0.3 |
| Mambo CMS | 2256 | -0.62 | 0.22 | -1 | 0 | 98% | -0.92 | 0.0 |
| MySQL | 19874 | -0.08 | 0.05 | 0.1 | -1 | 97% | 0.16 | -0.4 |
| OpenSSL | 1995 | 0.51 | 0.13 | 0.3 | -1 | 99% | 0.45 | -0.3 |
| PHPNuke | 816 | 0.94 | 0.11 | 0.9 | -1 | 98% | 0.75 | 0.9 |
| Plone | 1289 | 0.99 | 0.25 | 0.9 | 0 | 98% | 0.75 | -0.3 |
| postgresql | 2916 | -0.7 | 0.13 | -0.9 | -1 | 97% | -0.81 | 0.7 |
| Servicemix | 335 | -0.04 | 0.03 | 0.2 | 0 | 99% | 0.31 | 0.2 |
| Subversion | 335 | -0.04 | 0.03 | 0.2 | 0 | 99% | 0.31 | 0.2 |
| Subversion SVN | 1730 | 0.28 | -0.02 | 0.2 | -1 | 99% | 0.24 | 0.4 |
| Talend | 575 | -0.26 | 0.18 | -0.5 | -1 | 97% | -0.64 | -1.0 |
| Weka | 553 | 0.45 | -0.27 | 0.1 | 0 | 98% | 0.18 | 0.5 |
| Xerces | 778 | -0.59 | -0.09 | -0.6 | -1 | 99% | -0.82 | 0.3 |
| Xoops | 2261 | 0.67 | -0.11 | 0.6 | 0 | 98% | 0.76 | -0.4 |
| ZFS | 3874 | -0.29 | 0.22 | -0.6 | -1 | 97% | -0.76 | -0.1 |

Table 3: Number of posts collected and polarity analysis.

| | Opin1 | | OpinSVM | | NLTK | | Vivekl | | Senti Strength | | Senti Rank | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Prec. | Rec. | Prec. | Rec. | Prec. | Rec. | Prec. | Rec. | Prec. | Rec. | Prec. | Rec. |
| Apache httpd server | 0.69 | 0.49 | 0.70 | 0.52 | 0.17 | 0.05 | 0.33 | 0.11 | 0.17 | 0.05 | 0.61 | 0.34 |
| BusyBox | 0.94 | 0.62 | 0.94 | 0.63 | 0.89 | 0.35 | 0.83 | 0.20 | 0.56 | 0.05 | 0.33 | 0.02 |
| CVS | 0.76 | 0.57 | 0.78 | 0.64 | 0.52 | 0.19 | 0.40 | 0.12 | 0.57 | 0.24 | 0.73 | 0.48 |
| DDD data display debugger | 0.68 | 0.42 | 0.73 | 0.55 | 0.20 | 0.05 | 0.66 | 0.38 | 0.56 | 0.25 | 0.74 | 0.57 |
| DotNetNuke | 0.84 | 0.55 | 0.81 | 0.46 | 0.76 | 0.34 | 0.57 | 0.14 | 0.73 | 0.29 | 0.81 | 0.45 |
| Drupal | 0.62 | 0.49 | 0.69 | 0.68 | 0.06 | 0.02 | 0.06 | 0.02 | 0.72 | 0.77 | 0.63 | 0.51 |
| eclipse TPTP | 0.68 | 0.84 | 0.71 | 0.70 | 0.42 | 0.21 | 0.44 | 0.22 | 0.49 | 0.27 | 0.51 | 0.30 |
| eZ publish | 0.79 | 0.77 | 0.80 | 0.83 | 0.61 | 0.32 | 0.35 | 0.11 | 0.46 | 0.18 | 0.58 | 0.30 |
| GNU Project Debugger | 0.79 | 0.81 | 0.75 | 0.66 | 0.39 | 0.14 | 0.30 | 0.09 | 0.33 | 0.11 | 0.33 | 0.11 |
| jasper report | 0.69 | 0.49 | 0.77 | 0.72 | 0.42 | 0.16 | 0.00 | 0.00 | 0.32 | 0.10 | 0.57 | 0.29 |
| JBoss | 0.72 | 0.99 | 0.74 | 0.82 | 0.34 | 0.15 | 0.36 | 0.16 | 0.25 | 0.10 | 0.43 | 0.22 |
| JMeter | 0.83 | 0.31 | 0.90 | 0.58 | 0.20 | 0.02 | 0.90 | 0.58 | 0.86 | 0.37 | 0.92 | 0.69 |
| Joomla | 0.71 | 0.82 | 0.72 | 0.89 | 0.37 | 0.20 | 0.11 | 0.04 | 0.37 | 0.20 | 0.47 | 0.30 |
| linux debian | 0.64 | 0.52 | 0.74 | 0.83 | 0.41 | 0.20 | 0.41 | 0.20 | 0.56 | 0.36 | 0.49 | 0.27 |
| Mambo CMS | 0.65 | 0.78 | 0.63 | 0.70 | 0.28 | 0.16 | 0.13 | 0.06 | 0.40 | 0.28 | 0.52 | 0.46 |
| MySQL | 0.53 | 0.97 | 0.53 | 0.97 | 0.00 | 0.00 | 0.49 | 0.82 | 0.48 | 0.79 | 0.59 | 1.24 |
| OpenSSL | 0.70 | 0.49 | 0.74 | 0.62 | 0.67 | 0.43 | 0.82 | 0.96 | 0.78 | 0.74 | 0.78 | 0.77 |
| PHPNuke | 0.95 | 0.77 | 0.94 | 0.69 | 0.78 | 0.15 | 0.20 | 0.01 | 0.67 | 0.09 | 0.83 | 0.20 |
| Plone | 0.68 | 0.77 | 0.69 | 0.80 | 0.31 | 0.17 | 0.46 | 0.32 | 0.19 | 0.08 | 0.21 | 0.10 |
| postgresql | 0.84 | 0.30 | 0.90 | 0.53 | 0.64 | 0.10 | 0.33 | 0.03 | 0.33 | 0.03 | 0.90 | 0.54 |
| Servicemix | 0.84 | 0.67 | 0.84 | 0.67 | 0.47 | 0.11 | 0.63 | 0.21 | 0.36 | 0.07 | 0.63 | 0.21 |
| Subversion SVN | 0.85 | 0.42 | 0.90 | 0.70 | 0.74 | 0.22 | 0.68 | 0.16 | 0.67 | 0.15 | 0.70 | 0.18 |
| Talend | 0.58 | 0.72 | 0.78 | 0.89 | 0.33 | 0.13 | 0.05 | 0.01 | 0.44 | 0.20 | 0.10 | 0.03 |
| Weka | 0.65 | 1.02 | 0.64 | 1.00 | 0.35 | 0.30 | 0.19 | 0.13 | 0.29 | 0.23 | 0.13 | 0.08 |
| Xerces | 0.60 | 0.42 | 0.70 | 0.68 | 0.32 | 0.14 | 0.26 | 0.10 | 0.66 | 0.56 | 0.68 | 0.61 |
| Xoops | 0.79 | 0.44 | 0.87 | 0.79 | 0.59 | 0.17 | 0.59 | 0.17 | 0.00 | 0.00 | 0.53 | 0.13 |
| ZFS | 0.67 | 0.64 | 0.74 | 0.90 | 0.52 | 0.34 | 0.43 | 0.23 | 0.12 | 0.04 | 0.26 | 0.11 |
| All Projects | 0.73 | 0.63 | 0.76 | 0.72 | 0.43 | 0.17 | 0.40 | 0.20 | 0.45 | 0.24 | 0.55 | 0.35 |

Table 4: Precision and Recall for each project.