2017 IEEE 25th International Requirements Engineering Conference

A Domain-Independent Model for Identifying Security Requirements

Nuthan Munaiah, Andrew Meneely, and Pradeep K. Murukannaiah
Department of Software Engineering, Rochester Institute of Technology, Rochester, NY 14623-5608, USA
Email: {nm6061,axmvse,pkmvse}@rit.edu

***Abstract*—Existing work on identifying security requirements relies on training binary classification models using domain-specific data sets to achieve a high accuracy. Considering that domain-specific data sets are often not readily available, we propose a domain-independent model for classifying security requirements based on two key ideas. First, we train our model on the description of weaknesses from the Common Weakness Enumeration (CWE) data set. Although CWE does not describe requirements, it describes security weaknesses that are manifestations of unrealized security requirements. Second, we exploit a one-class classification model that relies only on positive samples (description of weaknesses in CWE), eliminating the need for negative samples, collecting which can be nontrivial.**

We evaluated our model on three industrial requirements documents from different domains. We found that a One-Class Support Vector Machine trained with domain-independent CWE data set outperforms a model from prior literature by identifying security requirements with an average precision, recall and F-score of 67.35%, 70.48% and 67.68%, respectively. Further, considering data sets from prior literature (consisting of both positive and negative examples), we found that one-class classifiers trained with only positive examples outperformed binary classifiers trained with both positive and negative examples in two out of three evaluation data sets, demonstrating the potential value of one-class classification for security requirements identification.

***Index Terms*—security requirements; classification; common weakness enumeration; domain-independent; one-class svm**

I. Introduction

Understanding security requirements is a key step toward engineering a secure system. Accordingly, there has been a substantial amount work on security requirements engineering [1], with a strong focus on eliciting and analyzing security requirements [2]. For example, methods such as secure $i*$ [3] and misuse cases [4] provide systematic steps to elicit, represent, and reason about security requirements.

Software systems may sometimes be built with no systematic consideration of security, because, for example, security was initially not considered as critical to the system or security experts were not available during requirements engineering. Security requirements of such systems are likely to be incomplete, obscure and distributed across several textual documents. When such systems reconsider security (when a security vulnerability surfaces, for example), it is crucial to assess the extent to which the current requirements specify security. Manually inspecting requirements-related documents to identify security requirements can be tedious and error prone. Thus, there is a need for automated approaches to identify security requirements to aid secure software engineering.

Knauss et al. [5] show that security requirements can be automatically identified using a machine-learned *classifier* and that such a classifier performs best when trained with a domain-specific data set. Consider two industrial requirements documents from different domains: the Global Platform Specification (GPS) [6] and the Customer Premises Network (CPN) specification [7], for example; the results presented by Knauss et al. show that a Naïve Bayes classifier trained with and tested on GPS yields an F-score as high as 96%, but, when trained with CPN and tested on GPS, the F-score is as low as 23%.

Domain-specific training data sets are not readily available for most domains. Further, curating such a training data set, wherein an expert must classify requirements as security and non-security for a given domain, is time consuming. Thus, relying on domain-specific datasets limits the practical utility of automated security requirements identification.

In this work, we attempt to address a key question: *Can we train a domain-independent classifier that can effectively identify security requirements across domains?*

The first challenge in realizing a domain-independent security requirements classifier is to find a domain-independent training data set. However, we found no such data sets in the *problem* (requirements) domain. We then expanded our search beyond the problem domain to also include *solution* domains (e.g., descriptions of architectural tactics, weaknesses and vulnerabilities), which led us to the Common Weakness Enumeration (CWE) [8] as a potential training data set. The CWE is a formal list of software weakness types intended to serve as the common language for describing software security weaknesses in architecture, design, or code. Intuitively, software weaknesses could be considered as describing repercussions of unrealized security requirements. We conjecture that the language used to describe security requirements and that used to describe weaknesses overlap and thus it may be possible to train a domain-independent security requirements classifier using the description of weaknesses. For instance, the requirement "When applicable a Security Domain shall Verify the load file data block signature when requested by the OPEN." from the GPS [6] data set overlaps with the description of CWE-347 [9] "The software does not verify, or incorrectly verifies, the cryptographic signature for data." We explore this conjecture in the following research question:

RQ1 How effective is the CWE data set for training a domain-independent security requirements classifier?

2332-6441/17 $31.00 © 2017 IEEE
DOI 10.1109/RE.2017.79

The second challenge in realizing a domain-independent security requirements classifier is to find a specific classifier to train. Most classifiers, in their simplest forms, perform *binary classification*, which requires both positive (security requirements) and negative (non-security requirements) samples for training. However, the CWE data set provides only positive samples. Curating negative samples is nontrivial—there can be a variety of requirements not related to security. Also, training a binary classifier with only a few arbitrary negative samples may overfit the classifier to the training data, which does not not generalize for new, out-of-training, samples.

Due to the lack of a representative set of negative samples, we consider an alternative type of machine learning approach known as *one-class classification* [10], traditionally used for novelty and outlier detection. In the absence of negative samples, a one-class classifier learns the boundary of positive samples and treats samples outside the learned boundary as negative. In RQ2, we apply one-class classification to replicate the empirical analysis of Knauss et al. [5] to assess if the effectiveness of classification is an artifact of the training data set or the machine learning approach used.

RQ2 How effective is one-class classification for identifying security requirements?

Contributions: We propose (1) a domain-independent model trained using descriptions of weaknesses from the CWE data sets, offering a general solution to the security requirements identification problem; and (2) a one-class classification model as opposed to a binary classification model for identifying security requirements, thereby, eliminating the need to collect negative training samples. We believe that our contributions are novel and open new avenues to approach a variety of problems in security requirements engineering, in particular, and software engineering, in general.

II. Methodology

The following subsections describe the three key steps we follow to answer our research questions.

Step 1 Data sets collection
Step 2 Model building
Step 3 Performance evaluation

Step 1: Data Sets Collection

In the first step, we identified data sets to use in training and evaluating a model to classify security requirements. We manually identified and collected the training data sets, whereas, the evaluation data sets were publicly available [5]. We describe the approach followed to collect the training data sets and provide an overview of the evaluation data sets.

1) Training Data Sets: The training data sets collected in our work are an important contributions since there are no existing data sets containing domain-independent specification of security requirements, to the best of our knowledge.

We started the data sets collection with an intuition that descriptions of the repercussions of unrealized security requirements may be valuable for identifying security requirements. Two sources of information on such repercussions are: (1) the description of the vulnerabilities contained in the National Vulnerability Database (NVD) and (2) the summary and extended description of weaknesses in the Common Weakness Enumeration (CWE) database. We have chosen the latter, referred to as the CWE data set, in our study as the description of vulnerabilities tend to be at a lower level of abstraction.

The CWE data set collected in our work, in its simplest form, contains only the summary of the weaknesses. However, a variant of the CWE data set, called the Extended CWE (ExCWE) data set contains the summary of the weakness appended with the corresponding extended description. We chose to use two different variants of the CWE data set because the extended description is, as the name suggests, more verbose than the summary and may contain additional information that could be relevant.

The CWE data set is at a relatively higher level of abstraction as the description of weaknesses tends to be fairly generic in terms of the language used to describe the weakness and its applicability. For example, the vulnerability CVE-2015-6765 [11] is associated with the weakness CWE-416 which has the summary "Referencing memory after it has been freed can cause a program to crash, use unexpected values, or execute code." [12]. The extended description of CWE-416, on the other hand, is fairly detailed but still at a higher level of abstraction because it does not contain any code snippets.

We started the data collection by downloading the XML formatted data files (Version 2.10) from the CWE website [13]. We parsed the XML files to obtain the summary and extended description of all weaknesses, resulting in CWE and ExCWE data sets, each with descriptions of 720 weaknesses.

2) Evaluation Data Sets: We evaluate our domain-independent security requirements classifier via three expert-labeled data sets used in prior work [5]: Customer Premises Network (CPN) specification [7], Electronic Purse Specification (ePurse) [14] and Global Platform Specification (GPS) [6]. These data sets contain requirements specification in natural language classified by experts as security or non-security. Shown in Table I is the number of security and non-security requirements in the three evaluation data sets.

TABLE I
Distribution of security and non-security requirements in the evaluation data sets

Data Set	Number of Requirements	
	Security	Non-security
CPN	41	169
ePurse	83	41
GPS	63	115

Step 2: Model Building

In this step, we train one-class classification models with the CWE and ExCWE datasets, which contain descriptions of weaknesses in natural language. The first step toward training a classifier from these datasets is to represent the natural language descriptions in a vector space.

We used the Term Frequency-Inverse Document Frequency (TF-IDF) vectorizer [15] to construct a matrix in which each row represents a weaknesses, columns represent unique words (tokens) across all weaknesses and each entry in the matrix corresponds to the TF-IDF value of a token in a weakness. The rows in the TF-IDF matrix become training instances and the columns become features in the model.

The TF-IDF matrix typically contains several hundred columns. We performed feature selection by ordering the features by descending order of TF values and selecting the top n features. We built several models with varying values of n and selected the model that was most effective in identifying security requirements in the evaluation data sets. The top 100 tokens in the CWE training data set is shown in Figure 1 to enable an intuitive interpretation of the data set.

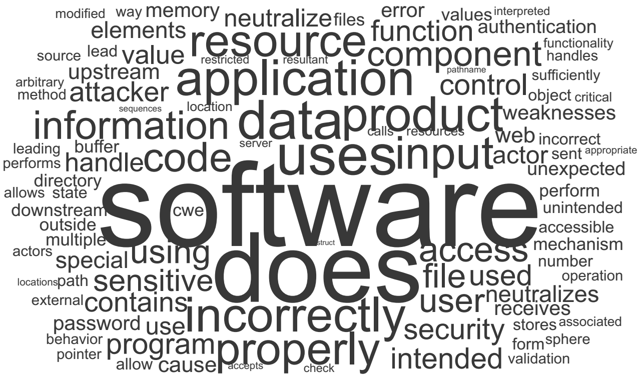

Fig. 1. Top 100 tokens in the CWE data set by cumulative TF-IDF scores

With the data sets in vector space, we trained One-Class Support Vector Machine (SVM) classifiers as implemented in scikit-learn [16], [17]. We used the Radial Basis Function as a kernel with the One-Class SVM. The radial basis function has a parameter, γ, that controls the influence of training samples on the support vectors. We tuned γ using 10-fold cross validated randomized parameter optimization [18] approach, repeated 100 times in each of the two training data sets. We found the optimal values to be $\gamma = 1.34\text{e}{-03}$ for CWE and $\gamma = 6.10\text{e}{-03}$ for ExCWE.

Step 3: Performance Evaluation

In this step, we assessed the performance of the one-class classification model when applied to identify security requirements in the evaluation data sets. We used the weighted-average versions of the standard binary classification performance evaluation metrics: precision (P), recall (R) and F-score (F_1). In weighted-averaging, the precision, recall and F-score are computed for each class (security and non-security in our case) and averaged by considering the number of true instances of each class to account for class imbalance.

III. Results

RQ1: How effective is the CWE data set for training a domain-independent security requirements classifier?

In this research question, we explore the effectiveness of using domain-independent data sets to identify security requirements. We use the one-class classification algorithm introduced earlier to build models by learning from the domain-independent data sets. We evaluate the performance of our models for each of the three evaluation data sets.

Figure 2 compares the performance metrics for One-Class SVM models trained with varying number of features from CWE and ExCWE data sets and tested on each of the three evaluation data sets. Shown in Table II are the performance metrics from the most effective One-Class SVM model in each of the three evaluation data sets. From Table II, we observe that the One-Class SVM models yield a mean F_1 score of 67.68%.

TABLE II
Performance of the most effective One-Class SVM model trained with the data sets proposed in our work

Data Set		Features	P	R	F_1
Evaluation	**Training**				
CPN	CWE	356	73.39%	79.52%	74.16%
ePurse	ExCWE	11	60.76%	64.51%	61.26%
GPS	ExCWE	11	67.90%	67.41%	67.62%

Shown in Table III are the performance metrics of the most effective models from Knauss et al. [5] in each of the three evaluation data sets. These models are based on Naïve Bayes classifiers trained with one or two of the three evaluation data sets and tested on one of the remaining data sets.

TABLE III
Performance of the most effective Naïve Bayes classifiers in prior work by Knauss et al. [5]

Data Set		P	R	F_1
Evaluation	**Training**			
CPN	GPS	65.00%	29.00%	40.00%
ePurse	GPS	48.00%	72.00%	58.00%
GPS	ePurse	85.00%	43.00%	57.00%

Comparing the metrics in Tables II and III, we observe that our one-class models outperform models from prior literature.

> One-Class SVM models trained using CWE data sets are effective at identifying security requirements. They yield a mean precision, recall and F-score of 67.35%, 70.48% and 67.68%, respectively, and outperform Naïve Bayes classifiers from prior work [5].

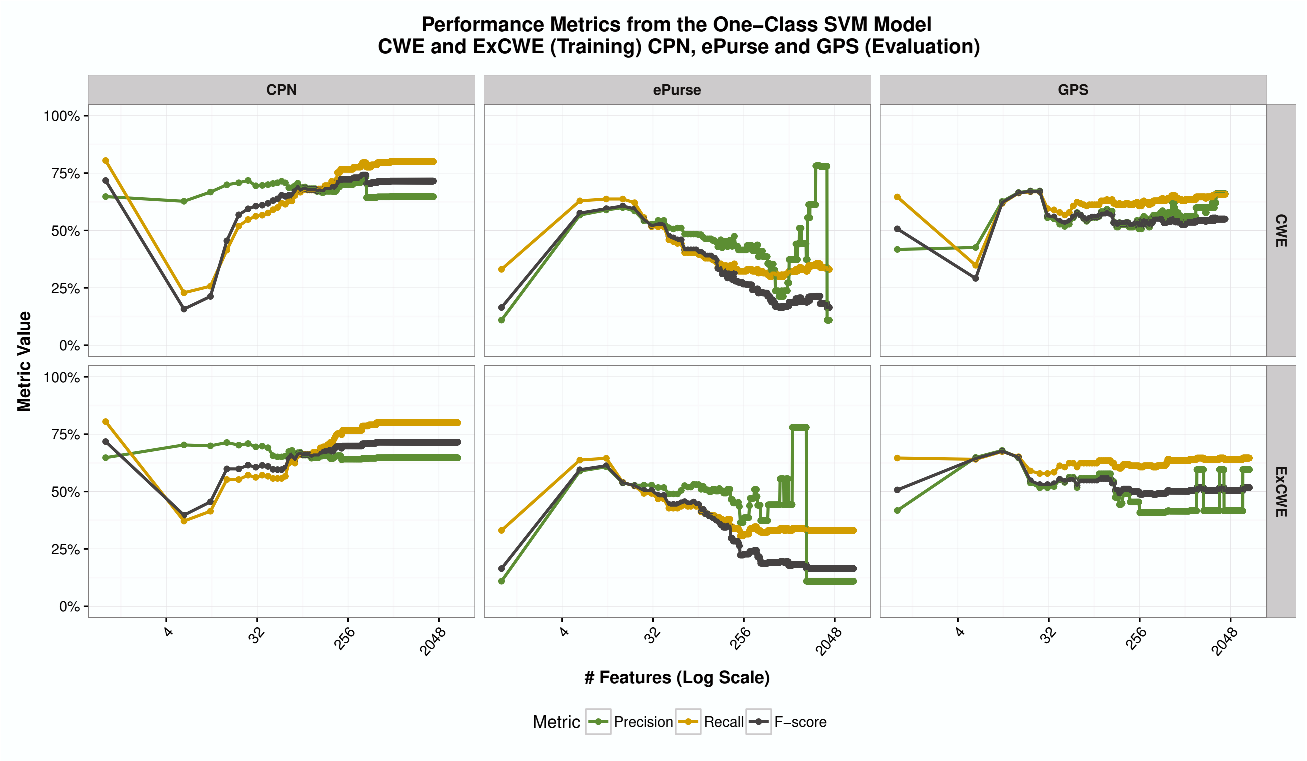

Fig. 2. Performance of the One-Class SVM model (trained on CWE and ExCWE data sets with varying number of features) when applied to identify security requirements in the evaluation data sets (CPN, ePurse and GPS)

RQ2: How effective is one-class classification for identifying security requirements?

In the previous research question, we found that one-class classification models trained with the CWE and ExCWE data sets are more effective in identifying security requirements than Naïve Bayes classifiers from prior literature. However, the performance of our models can be attributed to the training data sets or the one-class classification algorithm used to learn from the training data sets.

In this research question, we replicate a subset of the experiments from Knauss et al. [5] using the one-class classification approach to learn from security requirements in the training data sets. To enable a direct comparison, we replicated the experiments for the training and evaluation data set pairs from Table III. The performance metrics collected from the replication are presented in Table IV.

TABLE IV
PERFORMANCE OF THE ONE-CLASS CLASSIFICATION MODELS REPLICATED ON TRAINING AND EVALUATION SETS IN TABLE III

Data Set		**Features**	P	R	F_1
Evaluation	**Training**				
CPN	GPS	260	75.00%	80.47%	73.41%
ePurse	GPS	2	53.87%	50.81%	51.97%
GPS	ePurse	239	67.96%	67.41%	59.61%

Comparing Tables III and IV, we observe that the performance of the One-Class SVM model is better than that of the Naïve Bayes classifier from prior literature in two evaluation data sets, but worse for one evaluation data set. However, we note that the model trained with the domain-independent CWE data sets is still more effective than the one trained with the security requirements from the evaluation data sets.

> The One-Class SVM model is more effective at identifying security requirements than a Naïve Bayes classifier from prior work [5] in two of our three evaluation data sets.

IV. LIMITATIONS

We identify four limitations of our work, and directions for future research, where appropriate.

First, we considered two potential sources of data (Section II) that may be similar to descriptions of security requirements. However, we chose to use only one source—the description of weaknesses from the CWE—in the empirical analysis as the CWE data was at a higher level of abstraction than the other data set—the description of vulnerabilities from the NVD. As an exploratory exercise, we trained a One-Class SVM model using the NVD data set but the size of the NVD data set (84,161 rows) posed a challenge in collecting the model performance metrics across a wide range of features in the model. A future direction is to explore the suitability of NVD data set for security requirements identification.

Second, we studied only a single one-class classification algorithm, the One-Class SVM. We performed a preliminary exploration of the Isolation Forest [19], an alternative one-class classification approach. However, we found its performance to be worse compared to the One-Class SVM model for the subset of tests we performed. Isolation Forest and also other one-class classification approaches [20] remain vastly unexplored for security requirements identification (and for solving other problems in software engineering, in general).

Third, our domain-independent data sets and the One-Class SVM model go hand-in-hand. Indeed, the choice of using a one-class classification model was motivated by the fact that the CWE data sets provided us instances belonging to a single class (security requirements) only. As a result, we could not objectively evaluate the contributions of the data sets and the one-class classification algorithm to the performance gains that our models yield. However, we reiterate that our second research question is essentially controlling for the effect of the training data sets collected in our work while assessing the effectiveness of the classification algorithm alone.

Finally, although the One-Class SVM model we proposed outperforms the Naïve Bayes classifier from prior literature, the mean F-score (67.68%) of the One-Class SVM model leaves a lot to be desired. We believe that the utility of our models is in its ability to act as a preliminary sieve to identify security requirements. The subset of requirements identified by the model may then be manually assessed, considerably lowering the effort required had the model not been available.

V. Related Work

Understanding software requirements is a nontrivial task as requirements tend to be scattered across a variety of structured and unstructured sources. This also applies to, and the scattering is probably worse, for security requirements. Researchers have been attempting to aid designers and developers in understanding requirements by consolidating requirements from multiple sources [21] and identifying and visualizing quality concerns from requirements [22]. Since requirements are typically specified using natural language, researchers have attempted to classify requirements using natural language processing techniques [23], [24], [5].

At a high-level, requirements may be classified into functional and non-functional. Among the non-functional requirements, requirements related to security are regarded as critical as these requirements tend to be cross-cutting. Therefore, identifying security requirements is of prime importance. However, recruiting security experts to accomplish this goal may not be feasible. Fortunately, researchers [25], [5] have proposed classifiers to automatically identify security-related requirements from requirements specification.

A key challenge in building a classifier to identify different types of requirements is that the language used to specify requirements tends to vary across domains. The performance of the classifiers evaluated by Knauss et al. [5] shows that a classifier trained to identify security-related requirements in a particular data set performs poorly when applied to identify security-related requirements in another data set.

In contrast to existing works, we attempt to build a domain-independent model for identifying security requirements using a one-class classification model trained on data collected from a repository of common security weaknesses.

VI. Conclusion

Automatically identifying security requirements, and thus, knowing what is missing, is key to engineering a secure system. To this end, we explored the possibility of building a domain-independent model capable of identifying security requirements. We composed training data sets from descriptions of security weaknesses from the Common Weakness Enumeration (CWE) website. We then trained a One-Class SVM model with the training data set and assessed its effectiveness on three evaluation data sets from prior literature.

We found that the One-Class SVM's mean precision, recall and F-score in identifying security requirements are 67.35%, 70.48% and 67.68%, respectively, and that it outperformed a domain-dependent Naïve Bayes classifier from prior literature. In a follow-up, we replicated the empirical analysis from a prior work using the One-Class SVM to understand if the improved security requirement identification performance was due to the CWE training data sets or the machine learning approach used. We found that the One-Class SVM trained with the CWE data sets were more effective than that trained using security requirements from the evaluation data sets. Thus, both One-Class SVM and the domain-independent CWE data sets add value in automatically identifying security requirements.

References

[1] A. Souag, R. Mazo, C. Salinesi, and I. Comyn-Wattiau, "Reusable knowledge in security requirements engineering: A systematic mapping study," *Requirements Engineering*, vol. 21, no. 2, pp. 251–283, Jun. 2016.

[2] B. Fabian, S. Gürses, M. Heisel, T. Santen, and H. Schmidt, "A comparison of security requirements engineering methods," *Requirements Engineering*, vol. 15, no. 1, pp. 7–40, Mar. 2010.

[3] L. Liu, E. Yu, and J. Mylopoulos, "Analyzing security requirements as relationships among strategic actors," in *Proceedings of the 3rd Symposium on Requirements Engineering for Information Security*. Raleigh, NC: CERIAS - Purdue University, 2002, pp. 4:1–4:14.

[4] G. Sindre and A. L. Opdahl, "Eliciting security requirements with misuse cases," *Requirements Engineering*, vol. 10, no. 1, pp. 34–44, Jan. 2005.

[5] E. Knauss, S. Houmb, K. Schneider, S. Islam, and J. Jürjens, "Supporting requirements engineers in recognising security issues," in *Proceedings of the 17th International Working Conference on Requirements Engineering: Foundation for Software Quality*. Essen, Germany: Springer-Verlag, 2011, pp. 4–18.

[6] E. Knauss, S. Houmb, K. Schneider, S. Islam, and J. Jürjens, "Global Platform Specification (GPS)," http://www.se.uni-hannover.de/pub/File/projekte/secreq/GPS.csv, Accessed: 04-12-2017.

[7] ——, "Customer Premises Network (CPN) Specification," http://www.se.uni-hannover.de/pub/File/projekte/secreq/CPN.csv, Accessed: 04-12-2017.

[8] MITRE, "Common Weakness Enumeration: A community-developed list of software weakness types," https://cwe.mitre.org, Accessed: 04-11-2017.

[9] ——, "CWE - CWE-347: Improper Verification of Cryptographic Signature (2.11)," http://cwe.mitre.org/data/definitions/347.html, Accessed: 06-23-2017.

[10] S. S. Khan and M. G. Madden, “One-class classification: taxonomy of study and review of techniques,” *Knowledge Engineering Review*, vol. 29, no. 3, pp. 345–374, 2014.

[11] US-CERT/NIST, “NVD - CVE-2015-6765,” https://nvd.nist.gov/vuln/detail/CVE-2015-6765, Accessed: 04-11-2017.

[12] MITRE, “CWE - CWE-416: Use After Free (2.10),” http://cwe.mitre.org/data/definitions/416.html, Accessed: 04-11-2017.

[13] ——, “CWE - CWE List Version 2.10,” https://cwe.mitre.org/data/xml/cwec_v2.10.xml.zip, Accessed: 04-11-2017.

[14] E. Knauss, S. Houmb, K. Schneider, S. Islam, and J. Jürjens, “Electronic Purse Specification (ePurse),” http://www.se.uni-hannover.de/pub/File/projekte/secreq/ePurse-selective.csv, Accessed: 04-12-2017.

[15] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, “sklearn.feature_extraction.text.TfidfVectorizer - scikit-learn 0.18.1 Documentation,” scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html, Accessed: 04-12-2017.

[16] ——, “Scikit-learn: Machine Learning in Python,” *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[17] ——, “sklearn.svm.OneClassSVM - scikit-learn 0.18.1 Documentation,” http://scikit-learn.org/stable/modules/generated/sklearn.svm.OneClassSVM.html, Accessed: 04-12-2017.

[18] J. Bergstra and Y. Bengio, “Random search for hyper-parameter optimization,” *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[19] F. T. Liu, K. M. Ting, and Z. H. Zhou, “Isolation Forest,” in *2008 Eighth IEEE International Conference on Data Mining*, Dec 2008, pp. 413–422.

[20] S. S. Khan and M. G. Madden, *A Survey of Recent Trends in One Class Classification*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 188–197. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-17080-5_21

[21] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, “The Detection and Classification of Non-Functional Requirements with Application to Early Aspects,” in *14th IEEE International Requirements Engineering Conference (RE’06)*, sep 2006, pp. 39–48.

[22] M. Rahimi, M. Mirakhorli, and J. Cleland-Huang, “Automated Extraction And Visualization Of Quality Concerns From Requirements Specifications,” in *2014 IEEE 22nd International Requirements Engineering Conference (RE)*, aug 2014, pp. 253–262.

[23] J. Cybulski and K. Reed, *Requirements Classification and Reuse: Crossing Domain Boundaries*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 190–210. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-44995-9_12

[24] A. P. Nikora, “Classifying Requirements: Towards A More Rigorous Analysis Of Natural-language Specifications,” in *16th IEEE International Symposium on Software Reliability Engineering (ISSRE’05)*, Nov 2005, pp. 10 pp.–300.

[25] J. Cleland-Huang, R. Settimi, X. Zou, and P. Solc, “Automated classification of non-functional requirements,” *Requirements Engineering*, vol. 12, no. 2, pp. 103–120, 2007. [Online]. Available: http://dx.doi.org/10.1007/s00766-007-0045-1