Machine Learning Based Prediction of Change Request Severity Level: Experimental Results

Author1
Affiliation1

Author2
Affiliation2

***Abstract*—In the context of Change Request (CR) systems, the severity level of a change request is considered a critical variable when planning software maintenance activities, indicating how soon a CR needs to be addressed. However, the severity level assignment remains primarily a manual process, mostly depending on the experience and expertise of the person who has reported the CR. In this paper, we present preliminary findings of ongoing research aimed to predict the severity level of a CR by analyzing its long description, using text mining techniques and Machine Learning (ML) algorithms. Best results were obtained with a classifier based on the Random Forest ML algorithm. MARIO: Esta conclusao esta desatualizada. This classifier can predict whether the severity level will change (F1-measure of 0.7877841) and if it will increase or decrease (F1-measure of 0.622619). Preliminary results have shown that its F1-measure was 0.4902217 when predicting the final severity level in an imbalanced data scenario, which is a value similar to the literature. NEW: However, according to statistical tests, the evaluated ML algorithms had similar performance. REMOVED: We have also shown that only the classical measurements may not help deciding if the ML approach will bring any benefit to the user, and have proposed an alternative measuring approach to address this issue. NEW: We have also shown that the use classical ML measurements available in the literature may not help deciding whether a particular ML approach will bring any benefit to the user, and have proposed an alternative measuring approach to address this issue.**

MARIO: Nao vamos citar FLOSS nem no abstract nem no titulo? MARIO: No final, converter o pdf para .rtf e rodar o verificador ortografico do Word. ***Index Terms*—software maintenance; change request systems; machine learning; random forest.**

I. Introduction

Change Request (CR) systems have played a major role in maintenance process in many software development settings, both in Closed Source Software (CSS) MARIO: Ver outras ocorrencias. Nao e close e sim closed. and in Open Source Software (OSS) scenarios. This is specially true in OSS, which is characterized by the existence of many of users and developers with different levels of expertise spread out around the world, who might create or be responsible for dealing with several CRs [1].

A user interacts with a CR system often through a simple mechanism called CR form. This form enables him to request changes, to report bugs or to ask for support in a software product [2]. Initially, he or she should inform a short description, a long description, a type (e.g. bug, new feature, REMOVED: improvementNEW: enhancement, and task) and an associated severity level (e.g. blocker, critical, major, minor and trivial). Subsequently, a development team member will review this request and, case it is not refused for some reason (e.g. request duplication), he or she will complete the information in CR form, indicating, for example, its priority and assigning the person responsible for the CR.

The severity level information is recognized as a critical variable in the equation to estimate a prioritization of CRs [3]. It defines how soon the CR needs to be addressed [4]. However, the severity level assignment remains mostly a manual process which relies only on the experience and expertise of the person who has opened the CR [1], [3], [4]. As a consequence, it is a process with high degree of subjectivity, and it may be quite error-prone.

The number of CRs in large and medium software OSS projects [5] is frequently very large. MARIO: Coloque numeros para exemplificar.... and could reach tens of thousand records. Severity level shifts throughout CR lifecycle may have an adverse effect on planning of maintenance activities. For example, the maintenance team could be assigned to address less significant CRs before most important ones. There has been reports MARIO: Refs?? of efforts to implement intelligent software assistants to help developers and maintenance personnel in defining more accurately the field values in a CR form. Currently, Machine Learning techniques have become a popular method to address this issue and there is quite a few publications in this area in the literature [1].

Machine Learning (ML) techniques have been successfully applied in solving real problems in many areas of knowledge, including those related to CR systems, such as duplication and assignment of CR [1]. However, the accuracy of ML algorithms may be affected by imbalanced datasets [6] —a recurring critical problem in CR repositories [7]. For example, more than 60% of CRs may have a “major” severity level. In addition to this problem, most publications are still focused in predicting the severity level of CRs and none of them have been implemented into popular tools like as Bugzilla, Jira and Redmine. [1]. Furthermore, many have used proprietary and/or not public ML algorithms. Therefore, there is still a clear need of advances in this knowledge area, specially broadening the reach of research questions and including more popular and open OSS and ML algorithms.

In this context, the general purpose of our research is to develop an intelligent ML based assistant to help developers and maintenance personnel in the OSS maintenance activities. In this current article, our specific goals are: MARIO: Rediscutir os goals. Parece-me mais: evaluate existing ML alg applied to CR...... and how well they outperforms a human user in predicting CR severity level.

G_1 : Evaluate the performance of traditional ML algorithms in the prediction of CR severity level;

G_2 : Indentify a suitable algorithm to perform such prediction in a scenario where imbalanced data is natural;

G_3 : Propose a new metric to compare the performance of the software ML system and the user, in predicting or assigning the final CR Level.

MARIO: Veja no artigo anterior como eu coloquei bold nos G1, G2 etc

In order to meet these goals, this research works with the following research questions, regarding CR severity level during its lifecycle:

RQ_1 : *Will the CR severity level change?*

RQ_2 : *Will the CR severity level increase, decrease or remain the same*?

RQ_3 : *What is the prediction for the final CR severity level?*

RQ_4 : *How ML predictions compare to user prediction?*

MARIO: Idem bold

The contributions of our research are:

- Indicate the performance of three ML algorithms in multi category classifiers on imbalanced scenario.
- Propose a new way to measure the performance of ML algorithms taking into account the user prediction.
- Extend published results to include new FLOSS, new open ML algorithms, and new CR Repositories.

MARIO: Rediscutir contribuicoes e toda a lista de goals, rqs etc.

The article is organized as follows. Section II presents related work that are relevant to our research. Section III provides the information background about CR systems, text mining and machine learning techniques necessary to understand our approach. Section IV describes our work. Section V presents final findings and discussion. Finally, Section VI present conclusions and future work.

II. Related Work

This section presents relevant articles in the area of mining open system repositories, aiming at extracting data and using ML techniques to predict several maintenance properties.

Menzies and Marcus [8] have developed a method, named SEVERIS (SEVERity ISsue assessment), for evaluating the severity of CRs. SEVERIS is based on established data and text mining techniques. The method was applied to predict CR severity level in five projects managed by the Project and Issue Tracking System (PITS), an issue tracker system used by NASA (Stratified F-measures by severity level in the range: (2) 78%-86%; (3) 68%-98%; (4) 86%-92%).

Lamkanfi et al. [4] have developed an approach to predict if severity of bug report is non-severe (severity levels: 1 or 2) or severe (severity levels: 4 or 5) based on text mining algorithms (tokenization, stop word removal, stemming) and on the Naïve Bayes machine learning algorithm. They have validated their approach with data from three open source project (Mozilla, Eclipse, and GNOME). The article reports that a training set with approximately 500 CRs per severity level is sufficient to make predictions with reasonable accuracy (precision and recall in the range 0.65-0.75 with Mozilla and Eclipse; 0.70-0.85 with GNOME).

Valdivia et al. [9] have characterized blocking bugs in six open source projects and proposed a model to predict them. Their model was composed of 14 distinct factors or features (e.g. the textual description, location the bug is found in and the people involved with the bug). Based on these factors they have build decision trees for each project to predict whether a bug will be a blocking bug or not (F-measures in the range 15-42%).

Tian et al. [3] have develop a method to predict the severity level of new CRs based on similar CRs reported in the past. The comparison between old and new CRs was implemented by the BM25 similarity function. This method was applied to Mozilla, Eclipse and OpenOffice projects over more than 250,000 CR extracted from Bugzilla (F-measure in the range 13.9-65.3% for Mozilla; 8.6-58% for Eclipse; and 12.3-74% for OpenOffice).

MARIO: Estou considerando que nao ha novidades nesta secao, certo? Talvez fosse melhor fazer uma busca bibliografica rapida pra ver se apareceu algo novo no ultimo ano. Filtrar por ano

III. Background

This section briefly comments of basic concepts necessary to REMOVED: comprehend NEW: understand this research area, namely CR Systems, Text Mining, Machine Learning, and ML evaluation metrics.

Data used in this research area are usually extracted from the so called CR Systems, or Bug Tracking Systems. Popular CR Systems are Bugzilla, Jira, and Redmine [3]. MARIO: Liste tambem os datasets que vc vai usar no artigo. Additional information can be found in [10].

Two techniques are frequently used in this research area: Text Mining [11] [12] and Machine Learning (ML) [12] [13] [14] [15]. Detailing of these techniques are outside the scope of this paper.

Finally, it is worth mentioning the specific metrics we use for assessing prediction performance. The three most common performance measures for evaluating the accuracy of classification algorithms are precision, recall, and F-measure, described as follows [16] [17]:

Recall. Recall is the number of True Positives (TP) divided by the number of True Positives (TP) and of False Negatives (FN), where the TP and FN values are derived from the confusion matrix. A low recall indicates many false negatives.

Precision. Precision is the number of True Positives (TP) divided by the number of True Positives and False Positives (FP). A low precision can also indicate a large number of false positives.

F-measure. F-measure conveys the balance between precision and recall, and can be calculated as their harmonic mean.

IV. Experiment

This section describes the experiment conducted to address the Research Questions. As in typical methodologies used in ML studies, it comprises the following steps: Data Collection (IV-A), Data Preprocessing (Section IV-B), and Training and Testing (Section IV-C).

A. Data Collection

This step in the experimental research encompasses selecting REMOVED: aFLOSS NEW: datasets to serve as the data source, studying and interpreting its data structure, and finally extracting relevant data from its repository (feature extraction). NEW: In this research, Cassandra, Hadoop, Linux, Mozilla, and Spark Open Systems were considered as potential Open Source Systems to study. In a first approximation, Cassandra, Hadoop and Spark were selected as data sources of CR records, due to the fact they are open, well stablished, have a considerable number of CRs already registered, use standard repositories, and were under study by other researchers in our research group. MARIO: Corrigi aqui varios erros de concordancia de numero.

NEW: According to [wikipedia.org], Cassandra is a free and open-source distributed NoSQL management system designed to handle large amounts of data across many commodity servers, providing high availability with no single point of failure. MARIO: Wikipedia nao eh uma referencia adequada aqui. Use a da empresa ou algum artigo que compare estes repositorios. NoSQL esta correto?. [hadoop.apache.org] describes HADOOP as a "framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models". It is considered a specialized and complex OSS project with many users with different levels of expertise. NEW: Finally, according to [wikipedia.org] MARIO: Idem Wikipedia Spark is an open-source cluster-computing framework which provides an interface for programming entire clusters with implicit data parallelism and fault-tolerance. MARIO: Use aqui paralelismo na descricao dos 3 sistemas. Algo do tipo: Sistema1 (ref) eh.... usado para.... e usa a ferramenta..... Do jeito que esta ficou bastante nao paralelo.

Its MARIO: nao seria their? De qualquer maneira, talvez seja melhor repetir aqui o nome dos 3. CR repositories allows for access to all CR contents in XML format. Everything is available (except change history), from CR long description field (with lines with few characters to ones with many lines), including code snippets and exception stack trace.

CRs in NEW: Cassandra, HADOOP and Spark are stored in a Jira based repository [https://www.atlassian.com/software/jira]. Two steps are used to perform data extraction from NEW: their web site[http://issues.apache.org]: (i) copying CR basic data (e.g. status and resolution) from XML contents; and (ii) copying CR changes history from external HTML pages (this may be important for learning). REMOVED: About 10% of the CRs collected in this process changed their level of severity at least once.

CR record data from February 01, 2006 to May 07, 2017 were collected. REMOVED: Only requests from the common module (identifier = HADOOP-*)were considered for retrieval. The total number of CR records retrieved after preprocessing was NEW: 22901.

MARIO: Tentar melhorar a legibilidade dos graficos. Por exemplo, na fig 1, os percentuais das fatias da pizza estao ilegiveis. MARIO: Colocar a legenda das figuras na figura. O texto esta correto mas eh incomodo. Por ex. fig1a distribuicao de severity (1=blocking.....), fig1b Change pattern distribution, fig1c severity went up or down.

NEW: Figure 1 shows how the 7538 retrieved CR records were distributed in terms of severity level and severity level change. Figure 1(a) shows the severity level distribution: 9.7% have severity 1 (trivial); 37.6% have severity 2 (minor); 48.4% have severity 3 (major), 3.0% have severity 4 (critical), and 1.3% have severity 5 (blocker). Figure 1(b) shows that only 7% have changed their severities levels during the CR lifecycle. Finally, Figure 1(c) reveals that of these 7.7% CRs which changed their severity, 67% decreased it, and 33% increased it. MARIO: Nao caberia um comentario sobre estes dados? Por que vc apresentou? Tipo, observem que o n. de mudancas eh relativamente baixo.

Figure 2 shows how the 8262 retrieved CR records were distributed in terms of severity level and severity level change. Figure 2(a) shows the severity level distribution: 4.3% have severity 1 (trivial); 19.6% have severity 2 (minor); 61.2% have severity 3 (major), 3.8% have severity 4 (critical), and 11.1% have severity 5 (blocker). Figure 2(b) shows that only 8.0% have changed their severities levels during the CR lifecycle. Finally, Figure 2(c) reveals that of these 8.6% CRs which changed their severity, 30.8% decreased it, and 69.2% increased it.

NEW: Figure 3 shows how the 7101 retrieved CR records were distributed in terms of severity level and severity level change. Figure 3(a) shows the severity level distribution: 2.9% have severity 1 (trivial); 4.4% have severity 2 (minor); 22.5% have severity 3 (major), 50.6% have severity 4 (critical), and 10.1% have severity 5 (blocker). Figure 3(b) shows that only 13.3% have changed their severities levels during the CR lifecycle. Finally, Figure 3(c) reveals that of these 13.3% CRs which changed their severity, 35.9% decreased it, and 64.1% increased it.

One can see that these datasets are clearly imbalanced, posing additional difficulty to the application of the ML methodology. MARIO: Este tipo de comentario que eu estava esperando. E, dada a formatacao igual das 3 figs, eh melhor concentrar os comentarios aqui. Estender para outras caracteristicas, por ex o baixo n. de mudancas.

B. Preprocessing

REMOVED: The raw data NEW: Raw data previously collected from the Cassandra, HADOOP and Spark CR repositories were MARIO: data eh plural, e nao se usa the no inicio da frase em substantivo plural not properly structured to serve as input to ML algorithms, it REMOVED: wasNEW:

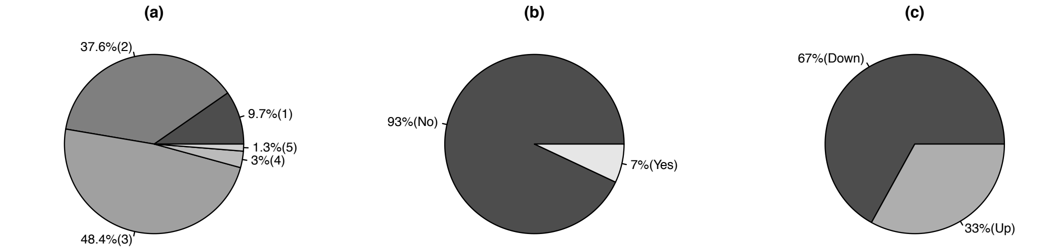

Fig. 1: Cassandra dataset distribution.

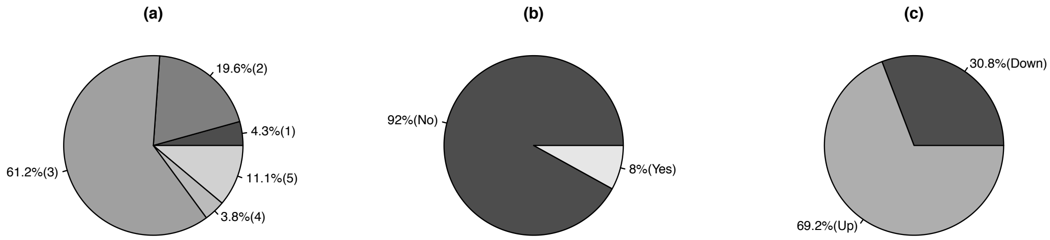

Fig. 2: Hadoop dataset distribution.

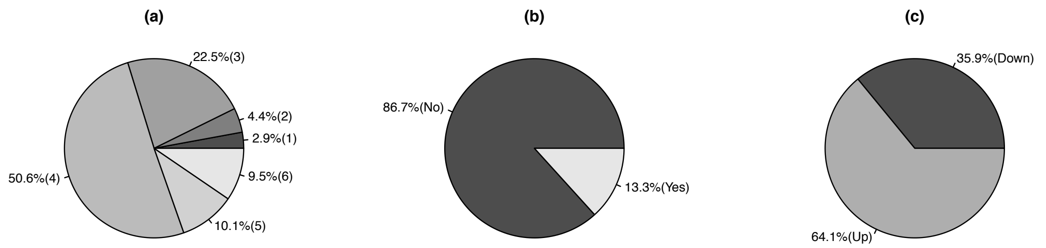

Fig. 3: Spark dataset distribution.

wasn't in tidy data format [18]. The classical way to address this problem is to run preprocessing procedures to extract, organize and structure relevant information out of the raw data. MARIO: Re-escrever para incluir a expressao feature, que me parece ser classica aqui. Um leitor experiente da area poderia se confundir achando que isso nao sao as features. Specific scripts were written in R language to accomplish this NEW: feature extraction. Preprocessing tasks were executed as follows:

- Extraction of relevant features: key, type, status, resolution status, and long description of CRs;
- REMOVED: Filtering NEW: Selecting only CRs with status equals to Closed and resolution equals to Fixed and Implemented.
- Merging CR features with their change history data. This additional information allows for the identification of CRs that have changed severity level during the CR lifecycle, and furthermore, if they have changed for better (decrease) or worse (increase).
- Performing text mining in the long description field to identify the 100 most frequent words. This information is then converted into features for each CR.
- Sorting the CRs in ascending chronological date.

MARIO: Espera-se uma justificativa para as decisoes tomadas aqui. Por exemplo, por que soh closed?

C. *Training and testing*

Training and testing steps start with partitioning the already preprocessed dataset in two disjoint subsets: a subset for training, with REMOVED: 60NEW: 80% of the CRs, and a subset for testing, with the remaining REMOVED: 40NEW: 20% of the CRs. Three classical sampling approaches, random, proportional, and uniform, MARIO: refs were analyzed to select the training set. Best results were obtained with the random sampling technique. In the training phase, we have used the REMOVED: 3-fold NEW: 5×3 Cross-Validation technique [17] REMOVED: to choose the best values for hyperparameters NEW: to obtain more stable estimates of each algorithm's performance and enhance replicability of the results [19]. REMOVED: for each classifier algorithm to use them in the test phase.NEW: In the testing phase, each ML algorithm was validated with 20% of each CR dataset to measure its accuracy

NEW: Initially, MARIO: Por que initially? we have chosen three traditional ML algorithms: Neural Networks [20], Random Forest [15] and Support Vector Machine(SVM) [21]. We have used respectively neuralnet(with Single Hidden Layer), randomForest(with two hundred trees per forest), and kernlab (with Radial Basis Function Kernel and multi-class classification) libraries of R language to implement them.

V. Findings and Discussions

MARIO: Faltou uma introducao para posicionar o leitor sobre o que e como sera o texto desta secao.

A. RQ1: Will the CR severity level change?

The RQ1 is a simple binary problem, i.e., a question whose answer is trueNEW: (class 1) or falseNEW: (class 0). NEW: Tables I, II and III show the performance NEW: per dataset of the classifiers to predict the response to this issue.

TABLE I: Neural Network Performance on RQ1.

	Class	Precision	Recall	F-measure
Neural Network				
Cassandra	0	0.9530591	0.9868924	0.9696807
	1	0.7241379	0.4144737	0.5271967
Hadoop	0	0.9530516	0.9780514	0.9653897
	1	0.6339286	0.4409938	0.5201465
Spark	0	0.9125249	0.9509669	0.9313493
	1	0.6162162	0.4634146	0.5290023
	Average	0.7988197	0.7057988	0.7404609

TABLE II: Random Forest Performance on RQ1.

	Class	Precision	Recall	F-measure
Random Forest				
Cassandra	0	0.9596013	0.9989077	0.9788600
	1	0.9740260	0.4934211	0.6550218
Hadoop	0	0.9498208	0.9930407	0.9709500
	1	0.8289474	0.3913043	0.5316456
Spark	0	0.9274406	0.9709945	0.9487179
	1	0.7640449	0.5528455	0.6415094
	Average	0.9006468	0.7334190	0.7877841

TABLE III: SVM Performance on RQ1.

	Class	Precision	Recall	F-measure
Support Vector Machine				
Cassandra	0	0.9581371	1.0000000	0.9786211
	1	1.0000000	0.4736842	0.6428571
Hadoop	0	0.9477157	0.9994647	0.9729026
	1	0.9830508	0.3602484	0.5272727
Spark	0	0.9134555	0.9986188	0.9541405
	1	0.9819820	0.4430894	0.6106443
	Average	0.9640569	0.7125176	0.7810730

We tested the classifiers with 4580 (20% of 22901) CRs: 4154 have changed their severity level, and 426 haven't changed their severity level. We can observe that the three classifiers performed very closely. However, the Random Forest classifier have achieved a F1-Measure NEW: average somewhat better than the two others. MARIO: O que eh o F1? As tabelas so mostram F-Measure. Colocar os numeros da comparacao no texto. MARIO: Idealmente, seria melhor mostrar estes dados em figuras que permitissem a comparacao visual. Conversemos sobre isto.

REMOVED: Regarding the Random Forest classifierNEW: In addition to the previous metrics, we have done one step further and investigated the classifiers performance relating to the number of hits and errors made in answer to RQ1. NEW: Figure 4 indicates that Neural Network accuracy was 91.113% (4173/4580), Random Forest accuracy was 92,969% (4258/4580) and SVM accuracy was 93,318% (4274/4580). These performance figures are better than others [4], [9] reported in the literature (see Table X). MARIO: Critico: refazer os graficos da fig4; um para RQ1 e outro para RQ2. Comparar o desempenho dos metodos lado a lado para cada RQ. Nao faz sentido comparar o desempenho de RQ1 com RQ2 como esta.

B. RQ2: Will the CR severity level increase, decrease or remain the same?

The RQ2 poses a problem more difficult than the previous question. It is a question with three possible responses related to severity level: NEW: (class -1) it has decreased; NEW: (class 0) it has remained; and NEW: (class 1) it has increased. NEW: Tables IV, V, and VI shows the performance of the classifiers to predict the response to this issue. MARIO: Mesmo comentario que em RQ1.

TABLE IV: Neural Network performance on RQ2.

	Class	Precision	Recall	F-measure
Neural Network				
Cassandra	-1	0.4393939	0.2989691	0.3558282
	0	0.9523305	0.9819771	0.9669266
	1	0.7333333	0.3928571	0.5116279
Hadoop	-1	0.2272727	0.1111111	0.1492537
	0	0.9549266	0.9753747	0.9650424
	1	0.6060606	0.5172414	0.5581395
Spark	-1	0.2500000	0.1500000	0.1875000
	0	0.9119788	0.9516575	0.9313957
	1	0.5555556	0.4518072	0.4983389
	Average	0.6256502	0.5367772	0.5693392

We tested the classifiers with 4580 (20% of 22901) CR. Only now, we have three predicting situations: 4154 haven't changed their severity level, 246 have increased their severity level, and 180 have decreased their severity level. We can observe in the NEW: Tables IV, V and VI which the classifiers also performed very closely as question 1. However, the Random Forest classifier have achieved F-measure somewhat better than the two others.

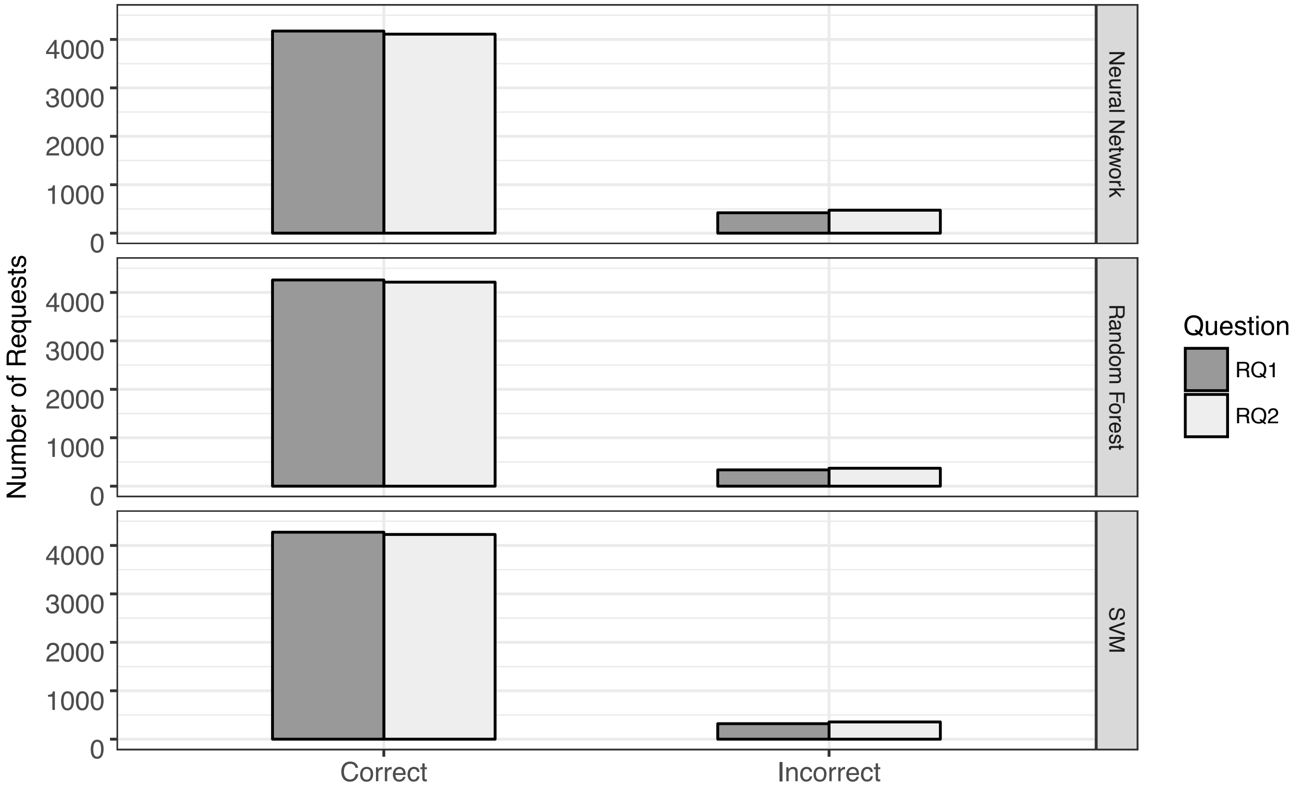

Fig. 4: Performance of classifiers for RQ1 and RQ2.

TABLE V: Random Forest performance on RQ2.

	Class	Precision	Recall	F-measure
Random Forest				
Cassandra	-1	0.7435897	0.2989691	0.4264706
	0	0.9565445	0.9978154	0.9767442
	1	0.7428571	0.4642857	0.5714286
Hadoop	-1	0.5555556	0.1111111	0.1851852
	0	0.9551084	0.9908994	0.9726747
	1	0.7195122	0.5086207	0.5959596
Spark	-1	0.5925926	0.2000000	0.2990654
	0	0.9303548	0.9779006	0.9535354
	1	0.6689655	0.5843373	0.6237942
	Average	0.7627867	0.5704376	0.6227619

TABLE VI: SVM performance on RQ2.

	Class	Precision	Recall	F-measure
Support Vector Machine				
Cassandra	-1	0.7631579	0.29896907	0.4296296
	0	0.9546403	1.00000000	0.9767938
	1	0.8214286	0.41071429	0.5476190
Hadoop	-1	0.5000000	0.08888889	0.1509434
	0	0.9520653	0.99946467	0.9751893
	1	0.8666667	0.44827586	0.5909091
Spark	-1	0.5769231	0.18750000	0.2830189
	0	0.9157695	0.99861878	0.5290023
	1	0.7977528	0.42771084	0.5568627
	Average	0.7942671	0.5400158	0.6073741

As in RQ1, we have done a step furtherREMOVED: regarding the best classifier. We have investigated its performance, observing the number of correct and incorrect answers on the test dataset as a whole. Figure 4 NEW: indicates NEW: that Neural Network accuracy was 89.716% (4109/4580), Random Forest accuracy was 91.986% (4213/4580) and SVM accuracy was 92.292% (4227/4580) in the RQ2 prediction. This performance is better than [3] and worse than [8] (see Table X).

C. *RQ3: What is the prediction for the final CR severity level?*

The RQ3 is a problem much harder than other two. It is a question with five responses related to severity level: (1) trivial; (2) minor; (3) major; (4) critical; and (5) blocker. NEW: Tables VII, VIII and IX shows the performance of the classifiers to predict the response to this issue. MARIO: Mesmo comentario que em RQ1 e RQ2.

We tested the classifiers with 4580 (20% of 22901) CRs. NEW: Only now, we have six predicting situations: 288 are trivial; 1218 are minor; 2470 are major; 259 are critical; 345

TABLE VII: Neural Network Performance on RQ3.

	Class	Precision	Recall	F-measure
Neural Network				
Cassandra	1	0.4022989	0.2243589	0.2880658
	2	0.5394737	0.5766526	0.5574439
	3	0.6645221	0.7053658	0.6843350
	4	0.4782609	0.3283582	0.3893805
	5	0.6666667	0.0800000	0.1428571
Hadoop	1	0.2000000	0.0131578	0.0246913
	2	0.3964497	0.1850828	0.2523540
	3	0.6668558	0.9136858	0.7709973
	4	0.3333333	0.1265822	0.1834862
	5	0.4032258	0.1111111	0.1742160
Spark	1	0.1818182	0.0312500	0.0533333
	2	0.3345070	0.2691218	0.2982731
	3	0.6096892	0.7494382	0.6723790
	4	0.4807692	0.3989361	0.4360465
	5	0.3703704	0.2531645	0.3007518
	Average	0.4485493	0.3310844	0.3485740

TABLE VIII: Random Forest Performance on RQ3.

	Class	Precision	Recall	F-measure
Random Forest				
Cassandra	1	0.6976744	0.1923077	0.3015075
	2	0.6209440	0.5921238	0.6061915
	3	0.6924959	0.8282927	0.7543314
	4	1.0000000	0.4626866	0.6326531
	5	1.0000000	0.2400000	0.3870968
Hadoop	1	0.9333333	0.1842105	0.3076923
	2	0.7433628	0.2320442	0.3536842
	3	0.7057175	0.9790047	0.8201954
	4	1.0000000	0.3544304	0.5233645
	5	0.9204545	0.3600000	0.5175719
Spark	1	0.7142857	0.0781250	0.1408451
	2	0.4785276	0.2209632	0.3023256
	3	0.6131657	0.9314607	0.7395183
	4	0.9733333	0.3882979	0.5551331
	5	0.7857143	0.2784810	0.4112150
	Average	0.7919339	0.4214952	0.4902217

are a blocker. We can observe in the Tables VII, VIII and IX which the classifiers also performed very closely as questions 1 and 2. However, the Random Forest classifier have achieved a F1-Measure somewhat better than the two others.

TABLE IX: SVM Performance on RQ3.

	Class	Precision	Recall	F-measure
Support Vector Machine				
Cassandra	1	0.5348837	0.1474359	0.2311558
	2	0.7513966	0.3783404	0.5032741
	3	0.6222510	0.9385366	0.7483469
	4	1.0000000	0.4626866	0.6326531
	5	1.0000000	0.2400000	0.3870968
Hadoop	1	0.8750000	0.1842105	0.3043478
	2	0.9452055	0.1906077	0.3172414
	3	0.6960305	0.9953344	0.8192000
	4	1.0000000	0.3417722	0.5094340
	5	1.0000000	0.3244444	0.4899329
Spark	1	0.8000000	0.0625000	0.1159420
	2	0.8194444	0.1671388	0.2776471
	3	0.5994532	0.9853933	0.7454314
	4	0.9726027	0.3776596	0.5440613
	5	0.9500000	0.2405063	0.3838384
	Average	0.8377511	0.4024377	0,4673068

D. RQ4: How ML predictions compare to user prediction?

We have compared NEW: classifiers predictions to user prediction in terms of error magnitude. Figure 5 (a) shows predictors versus user error magnitude in the assignment of severity level. Figure 5 (b) analyzes who had better performance (smaller error). MARIO: O que eh smaller error? Eh valor absoluto do erro? Deixar isto mais claro, escrevendo a equacao do erro. This type of measurement shows that the use of a software predictors results in no gain to the user. NEW: This conclusion could not be drawn simply knowing the value of the classic accuracy measurement for Neural Network(2426/4580 = 52.969%), Random Forest(2762/4580 = 60.305%), and SVM (2731/4580 = 59.628%). It is worth mentioning that this type of measurement, as reported in the literature, is the same range. MARIO: Discutir isto na reuniao. O raciocinio nao ficou claro. Therefore, one cannot state with confidence whether the use of the reported ML approach will bring any benefit, as compared to a simple educated guess by the user.

MARIO: Parei aqui.

E. Statistical Tests

We can one observe in the prior tables the performance of the three ML algorithms are very closely. To confirm if they are really equivalent, we have ran Friedman Test [19] using F1-measures input parameter. Table XI shows the null hypothesis(H0) can be accepted ($p\text{−value} < 0.05$) for the major of cases, confirming that our contexts the three algorithms investigated are equivalent [22].

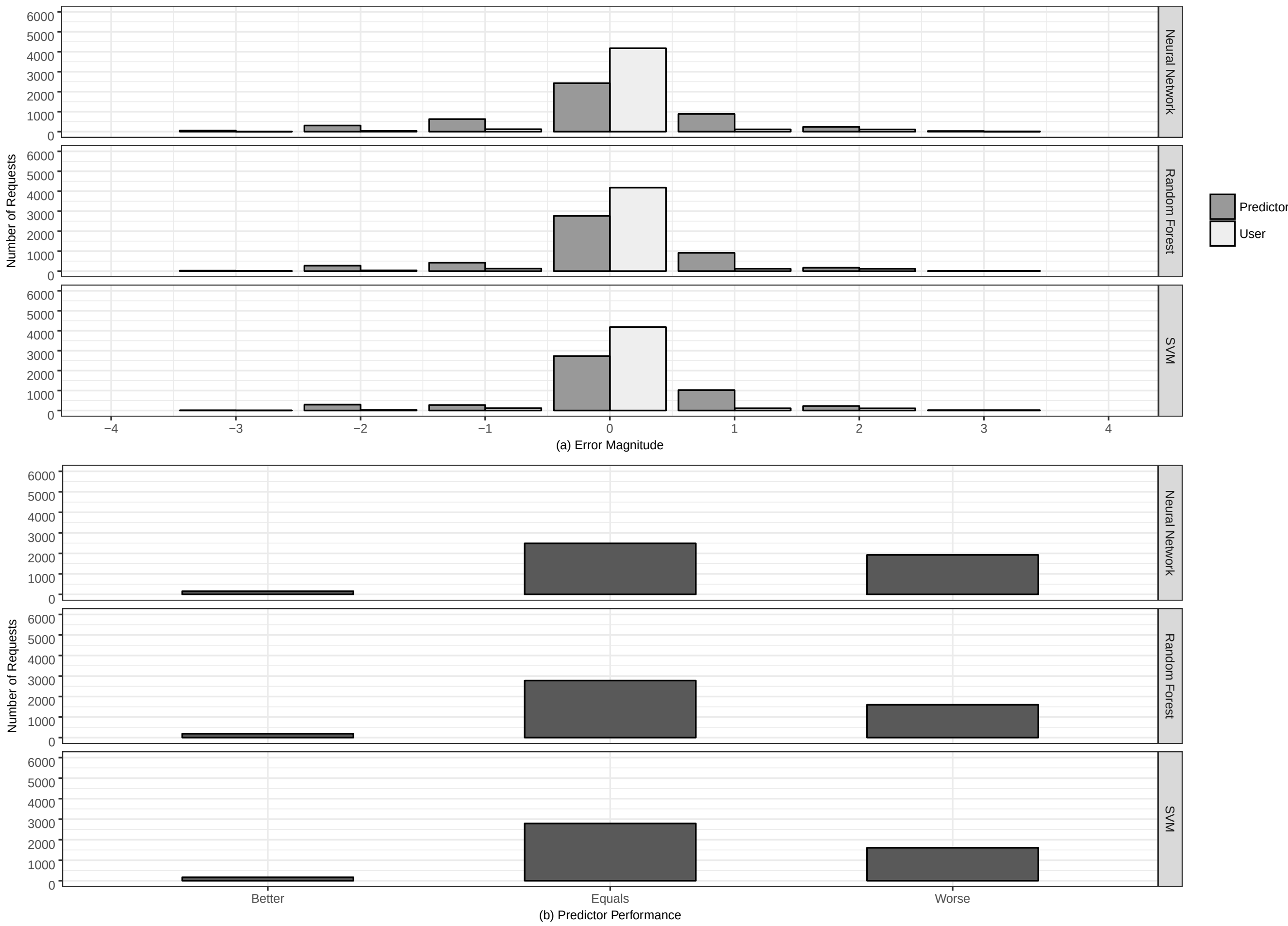

Fig. 5: Performance of classifiers for RQ3.

VI. Conclusions

In this paper, we have investigated the performance of three popular ML algorithms to predict CR severity level in an imbalanced data scenario. NEW: The results based on 22901 CRs extracted from the Cassandra, HADOOP e Spark repositories have shown that Random Forest had the somewhat better performance among the three algorithms to predict whether the severity level will change and whether it will increase or decrease with good F1-measure, around 0.7877841 and 0.622619 respectively, better than findings reported in the literature. However, it has provided the F1-measure (around 0.4902217) to predict the final severity level on imbalanced data scenario, which is similar to other results in the literature. NEW: However, according to Friedman Test, the three ML algorithms obtained equivalent performance in our context. We have also shown that the classical measurements do not help deciding if the ML approach will bring any benefit to the user, and have proposed an alternative measuring approach to address this issue.

Validity threats to our research are: (a) We have assumed that user assigned severity level is correct and that there is a close relationship between it and the long description of the CR. This assumption is supported [3], [4]. (b) NEW: We have considered one three repositories and we have extracted 22901 CRs from it. Although we can't generalize the results to others, the characteristics presented by Cassandra, HADOOP and Spark repositories, particularly regarding the balance of the data, are similar to those shown in the repositories studied [3]–[5], [9]. (c) Code developed in the Java language and the R language for preprocessing, training, testing and analysis of results have been carefully checked may still contain bugs.

As future work, we intend to investigate other repositories and systems, and develop an approach for representing CR Systems data in a general and uniform manner, so as to facilitate the development of a general purpose ML-based Maintenance Assistant.

Acknowledgment

(omitted for double-blind reviewing).

References

[1] Y. C. Cavalcanti, P. A. da Mota Silveira Neto, I. d. C. Machado, T. F. Vale, E. S. de Almeida, and S. R. d. L. Meira, "Challenges and opportunities for software change request repositories: a systematic mapping study," *Journal of Software: Evolution and Process*, vol. 26, no. 7, pp. 620–653, jul 2014.

[2] I. Sommerville, *Software Engineering*, 2010.

TABLE X: Classifiers Performance Summary.

	Research Questions	Projects	F-measure	Algorithms
Menzies [8]	Is the bug report blocker critical, major, minor trivial?	pitsA	14.0-71.0	RIPPER
		pitsB	42.0-90.0	RIPPER
		pitsC	53.0-92.0	RIPPER
		pitsD	87.0-99.0	RIPPER
		pitsE	8.0-80.0	RIPPER
Lamkanfi [4]	Is the bug report servere or non-severe?	Mozilla	65.9-71.7	Näive Bayes
		Elipse	62.5-65.5	Näive Bayes
		GNOME	72.7-78.5	Näive Bayes
Valdivia [9]	Is the bug report blocking or non-blocking?	Chromium	15.3	Decision Tree
		Eclipse	15.4	Decision Tree
		FreeDesktop	31.9	Decision Tree
		Mozilla	42.1	Decision Tree
		NetBeans	21.1	Decision Tree
		OpenOffice	25.6	Decision Tree
Tian [3]	Is the bug report blocker critical, major, minor trivial?	OpenOffice	12.3-74.0	INSPect
		Mozilla	13.9-65.3	INSPect
		Eclipse	8.6-58.6	INSPect
Ours	Will the CR severity level change?	HADOOP	96.7	Random Forest
	Will the CR severity level increase, decrease or remain the same?	HADOOP	19.6-96.6	Random Forest
	Is the bug report blocker, critical, major, minor, trivial?	HADOOP	9.3-79.9	Random Forest

TABLE XI: Friedman tests results over F-measure.

	Question	P-value	H0
Cassandra	Q1	0.135335283	Accepted
	Q2	0.096971968	Accepted
	Q3	0.055637998	Accepted
Hadoop	Q1	0.223130160	Accepted
	Q2	0.096971968	Accepted
	Q3	0.006737947	Reject
Spark	Q1	0.223130160	Accepted
	Q2	0.096971968	Accepted
	Q3	0.040762204	Reject

[3] Y. Tian, D. Lo, and C. Sun, “Information Retrieval Based Nearest Neighbor Classification for Fine-Grained Bug Severity Prediction,” in *2012 19th Working Conference on Reverse Engineering*, oct 2012, pp. 215–224.

[4] A. Lamkanfi, S. Demeyer, E. Giger, and B. Goethals, “Predicting the severity of a reported bug,” *Proceedings - International Conference on Software Engineering*, pp. 1–10, 2010.

[5] A. Lamkanfi, S. Demeyer, Q. D. Soetens, and T. Verdonckz, “Comparing mining algorithms for predicting the severity of a reported bug,” *Proceedings of the European Conference on Software Maintenance and Reengineering, CSMR*, pp. 249–258, 2011.

[6] N. V. Chawla, “Data Mining for Imbalanced Datasets: An Overview,” in *Data Mining and Knowledge Discovery Handbook*. Boston, MA: Springer US, 2009, pp. 875–886.

[7] Y. Tian, D. Lo, X. Xia, and C. Sun, “Automated prediction of bug report priority using multi-factor analysis,” *Empirical Software Engineering*, vol. 20, no. 5, pp. 1354–1383, oct 2015.

[8] T. Menzies and A. Marcus, “Automated severity assessment of software defect reports,” *IEEE International Conference on Software Maintenance, 2008. ICSM 2008*, pp. 346–355, 2008.

[9] H. Valdivia Garcia, E. Shihab, and H. V. Garcia, “Characterizing and predicting blocking bugs in open source projects,” *Proceedings of the 11th Working Conference on Mining Software Repositories - MSR 2014*, pp. 72–81, 2014.

[10] R. S. Pressman, *Software Engineering A Practitioner’s Approach 7th Ed - Roger S. Pressman*, 2009.

[11] R. Feldman and J. Sanger, *The Text Mining Handbook: Advanced Approaches in Analyzing Unstructured Data*. Cambridge University Press, 2007.

[12] G. Williams, *Data Mining with Rattle and R: The Art of Excavating Data for Knowledge Discovery*, 2011. [Online]. Available: http://books.google.com/books?id=mDs7OXj03V0C

[13] K. Surya, R. Nithin, and R. Venkatesan, “A Comprehensive Study on Machine Learning Concepts for Text Mining,” *International Conference on Circuit, Power and Computing Technologies [ICCPCT] A*, vol. 3, no. 1, pp. 1–5, 2016.

[14] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 2010.

[15] L. Breiman, “Random Forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[16] K. Facelli, A. C. Lorena, J. Gama, and A. Carvallho, *Inteligência Artifical: uma abordagem de aprendizado de máquina*. Rio de Janeiro: LTC, 2015.

[17] Y. Zhao and Y. Cen, *Data Mining Applications with R*, 1st ed. Academic Press, 2013.

[18] E. de Jonge and M. van der Loo, “An introduction to data cleaning with R,” *Statistics Netherlands*, p. 53, 2013. [Online]. Available: http://cran.r-project.org/doc/contrib/de_Jonge+van_der_Loo-Introduction_to_data_cleaning_with_R.pdf

[19] N. Japkowicz and M. Shah, *Evaluating Learning Algorithms: A Classification Perspective*. New York, NY, USA: Cambridge University Press, 2011.

[20] S. Haykin, *Neural Networks: A Comprehensive Foundation*, 2nd ed. Upper Saddle River, NJ, USA: Prentice Hall PTR, 1998.

[21] N. Cristianini and J. Shawe-Taylor, *An Introduction to Support Vector Machines: And Other Kernel-based Learning Methods*. New York, NY, USA: Cambridge University Press, 2000.

[22] J. Demšar, “Statistical comparisons of classifiers over multiple data sets,” *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Dec. 2006. [Online]. Available: http://dl.acm.org/citation.cfm?id=1248547.1248548