Using Source Code Metrics to Predict Change-Prone Web Services: A Case-Study on eBay Services

Lov Kumar[1] Santanu Kumar Rath[1] Ashish Sureka[2]

[1]NIT Rourkela, India
Email: lovkumar505@gmail.com, skrath@nitrkl.ac.in

[2]ABB India, India
Email: ashish.sureka@in.abb.com

MaLTeSQuE 2017

Table of Contents

Table of Contents

Maintainability Prediction - Service Oriented Software

Change-Prone Class Prediction

Focus preventive maintenance efforts such as testing, peer-reviews and source code refactoring on change-prone regions [7][10]

Web Services

Distributed web application components, implemented in different languages, deployed on different client/server platforms, represented by interfaces, communicate using open protocols [5][12]

WSDL (Web Service Description Language)

An XML format and provides users of the web service with a point of contact [5]

Research Gap & Unique Challenges

Change proneness prediction for web service interfaces

Service oriented systems is a different programming paradigm in comparison to object-oriented applications

Romano et al. mention that there are lack of studies that examine indicators of changes for service-oriented system [14]

Specific Research Aim

Can source code metrics be used to predict change-proneness of web services?

Table of Contents

Coscia et al. [4]

High correlation between several traditional (source code-level) OO metrics and the catalog of (WSDL-level) service metrics [4].

Lov et al. [8]

Quality of Service (QoS) characteristics of web services have a correlation with several source code metrics and hence can be estimated by analyzing the source code [8].

Several interface complexity attributes such as the data, relation, format, structure, data flow and language are correlated with source code implementing the web-services.

Table of Contents

Novel and Unique Contributions

Contribution on Solution Approach and Evaluation

First study on using source code metrics (implementing a web web-service) to predict change-prone web service interfaces (defined using WSDL) using kernel based learning techniques - Least Squares Support Vector Machines (LS-SVM).

Empirical evaluation of the proposed approach on real world and publicly available dataset consisting of several version of eBay services

Table of Contents

Experimental Dataset Details of eBay Web Services

Version	# Classes	LOC	# Changed	# Non-Changed	% Changed
863	1507	1320264	175	1332	11.61
865	1507	1320576	102	1405	6.77
867	1524	1335460	240	1284	15.75
869	1509	1322904	128	1381	8.55
871	1509	1323080	119	1390	7.89

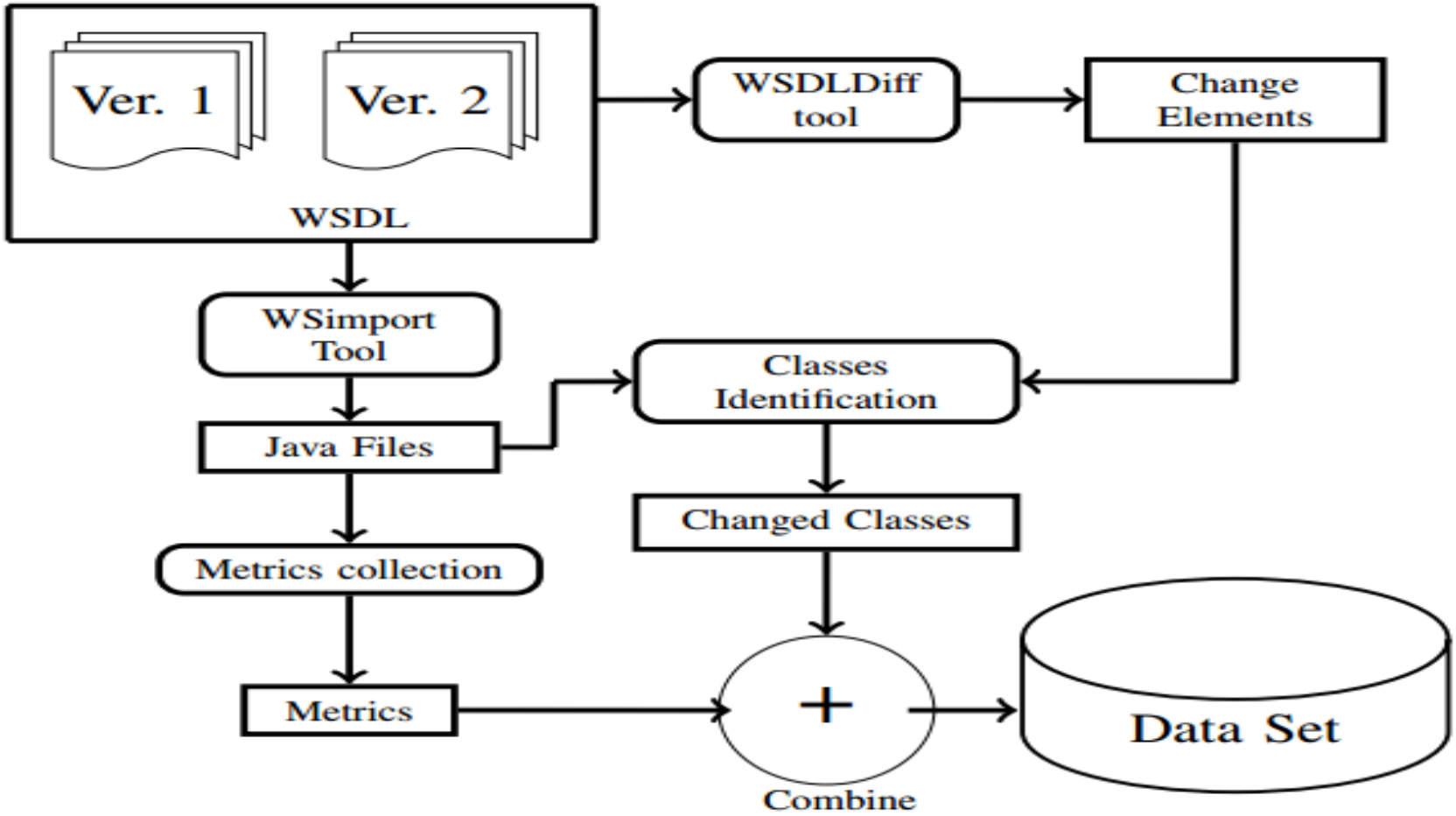
We download eBay web services from publicly available source[a] - our experiments can be replicated, used for benchmarking & comparison.

The eBay Trading API is an active project and consists of several versions. The latest version as of 17 June 2016 is 973.

[a] `http://developer.ebay.com/Devzone/XML/docs/ReleaseNotes.html`

Table of Contents

Multi-step Process of data Collection and Analysis



Data Processing Steps

We compute 21 source code metrics CKJM extended[a]

We use WSDL Diff tool for comparing subsequent versions of WSDL interfaces [3]. The WSDL Diff tool automatically extracts the changes in subsequent versions of WSDL interfaces.

We use the wsimport[b] tool to parse WSDL document file of a Web Service and generate its corresponding Java class (extracting the Java source code implementing the service)

---

[a] http://gromit.iiar.pwr.wroc.pl/p_inf/ckjm/

[b] http://docs.oracle.com/javase/6/docs/technotes/tools/share/wsimport.html

Data Processing Steps

The Java classes which contains changed (addition, deletion and modification) elements of the Web Service such as (Operation, Message, XSDType) are termed as changed classes.

The percentage of change classes from version 865 to 867 is 15.75 and the percentage of change classes from 869 to 871 is 7.89.

We consider all the changes as equivalent and do not differentiate between type of changes (a bug fix for a feature enhancement or refactoring) or the size of the change.

Two Research Questions

RQ1: Is it possible to predict change-proneness of web-services (defined using a WSDL document) using source code metrics implementing the web services? What is the performance of 21 CKJM metrics in-terms of their predictive power?

RQ2: What is the variation in performance (measured in-terms of accuracy and F-measure) of several LS-SVM classification models over different set of source code metrics? Do we see different performance for different (linear, polynomial and RBF) LS-SVM kernel functions?

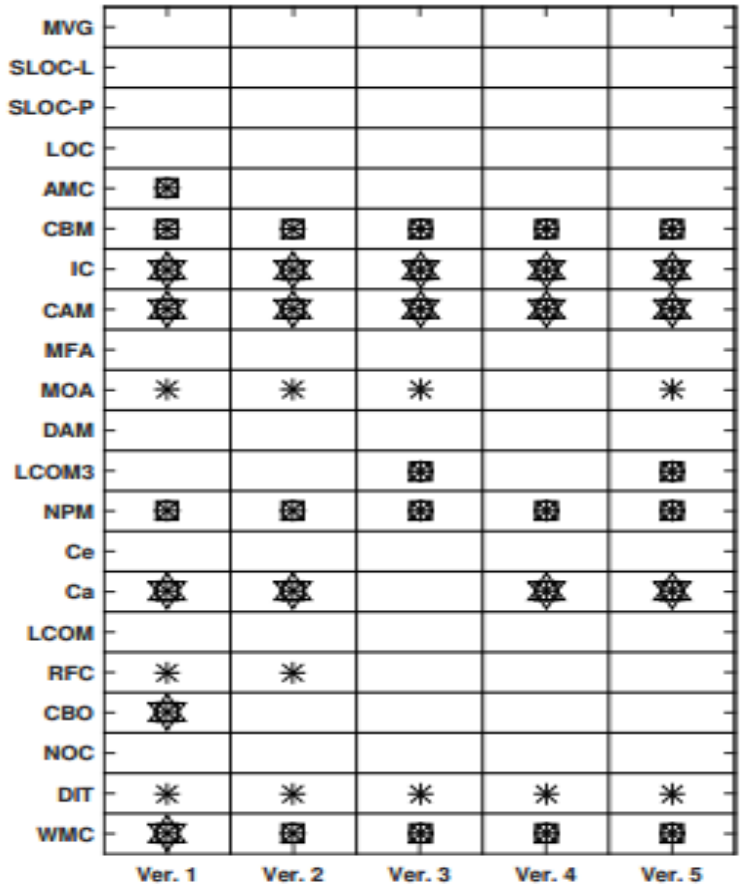
Table of Contents

Source code metrics - SM or Selected Metrics

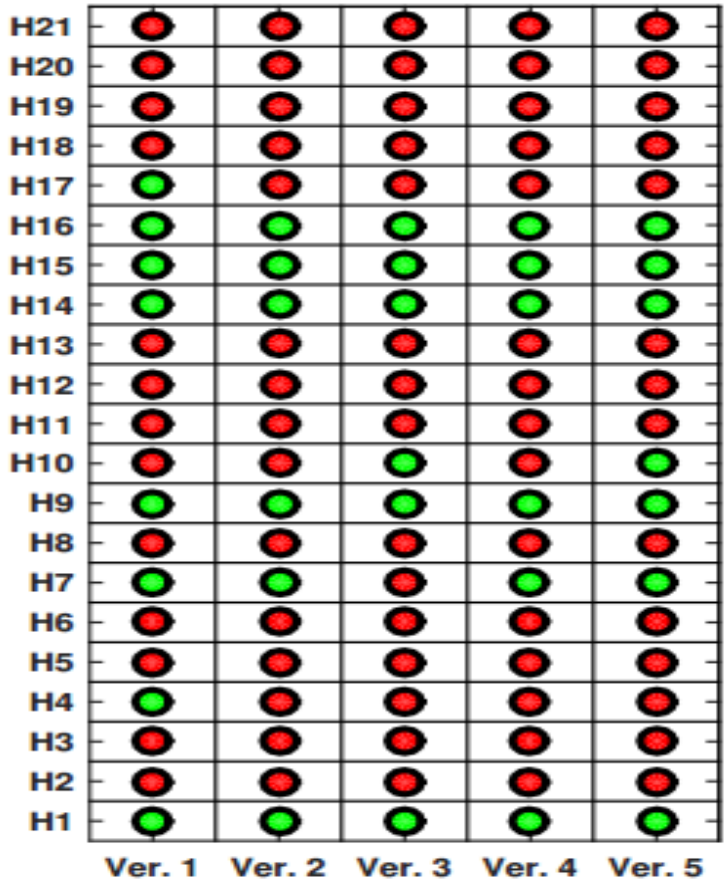
Four different symbols

1. Star ($*$): source code metrics selected after t-test analysis.
2. Circle with star ($\circledast$): source code metrics selected after t-test and ULR analysis.
3. Square with circle and star ($\boxtimes$): source code metrics selected after t-test, ULR analysis and cross correlation analysis.
4. Hexagonal with square, circle and star ($\hexagon$) : source code metrics selected after t-test, ULR analysis, cross correlation analysis and MLR stepwise forward selection method.

Source Code Metrics Validation



(a) Selected Set of Metrics



(b) Hypothesis

Source Code Metrics Validation

t-test Analysis

First step of source code metrics validation is to test the statistical significance between change and non-change-proneness group source code metric.

We apply $t - test$ on each source code metric and consider $p - value$ of each source code metrics.

The metrics having $p - value$ lesser than 0.05 have strong discriminatory power.

WMC, DIT, CBO, RFC, Ca, NPM, MOA, CAM, IC, CBM, and AMC source code metrics significantly differentiate the change or non-change-proneness classes for eBay version 863.

Source Code Metrics Validation

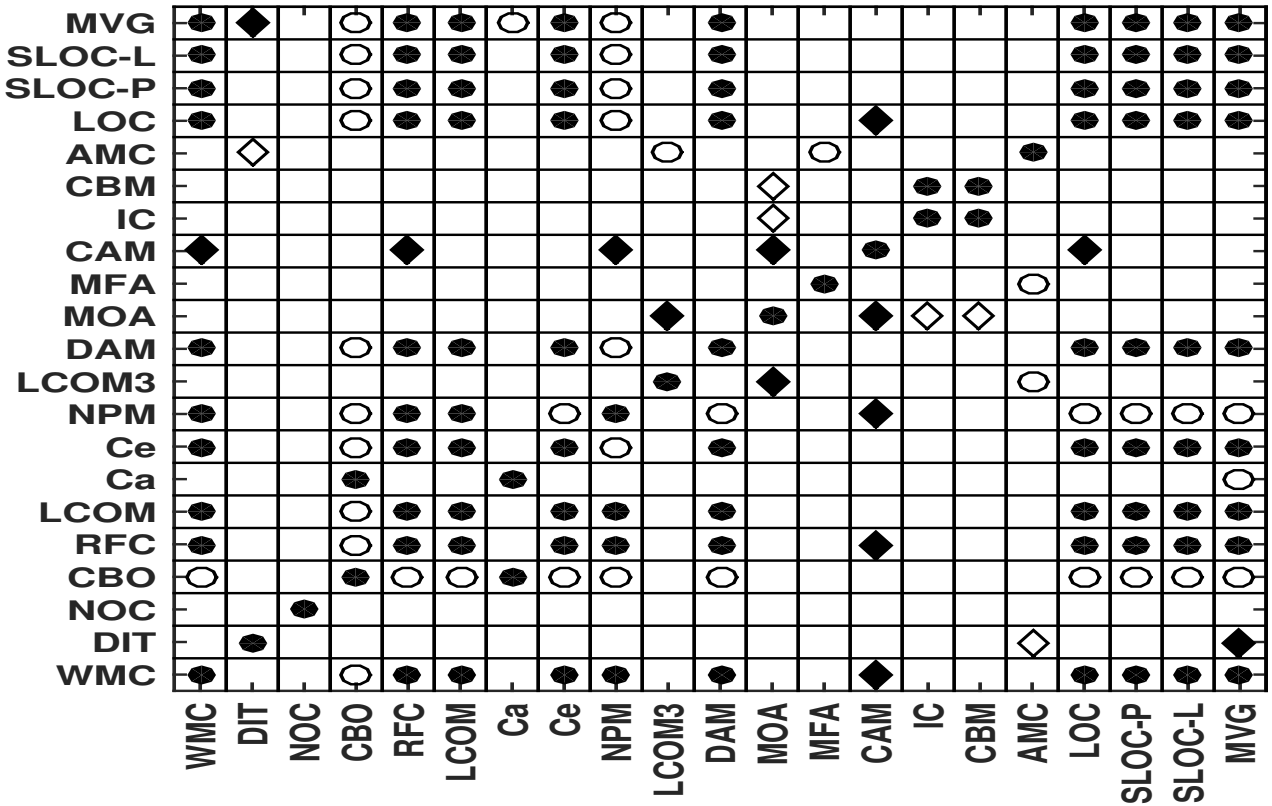
Univariate Logistic Regression (ULR) Analysis

Investigate whether the selected set of source code metrics using $t-test$ analysis are significant predictors of change-pronenes classes or not.

A source code metrics is significant predictor of class change-proneness if its p-value of coefficient is less than 0.05.

WMC, DIT, CBO, RFC, Ca, NPM, MOA, CAM, IC, CBM, AMC only WMC, CBO, Ca, NPM, CAM, IC, CBM, AMC source code metrics are significant predictors of change-proneness for eBay web service version 863.

Correlation Analysis between Metrics



Correlation Analysis between Metrics - Symbols

1. Black Circle (●): r value between 0.7 and 1.0 indicate a strong positive linear relationship.
2. White circle (○): r value between 0.3 and 0.7 indicate a weak positive linear relationship.
3. Black Diamond (): r value between -1 and -0.7 indicate a strong negative linear relationship.
4. White Diamond (◇): r value between -0.7 and -0.3 indicate a weak negative linear relationship.
5. Blank Circle: no linear relationship.

Multivariate Linear Regression Stepwise Forward Selection

Multivariate Linear Regression Stepwise Forward Selection:

The selected set of source code metrics obtained after cross correlation analysis does necessarily mean that we have a suitable set of source code metrics for change-proneness model development.

Multivariate linear regression stepwise forward selection method is considered to select right set of source code metrics for development of change-proneness models.

Table of Contents

Least Square Support Vector Machine (LSSVM) Classifier

LSSVM

LSSVM classifier with three different kernel methods for validating the selected set of source code metrics.

20 fold cross-validation for comparing the predictive models.

The performance of each prediction model is evaluated in terms of two performance parameters i.e., accuracy and F-Measure.

Performance Parameters

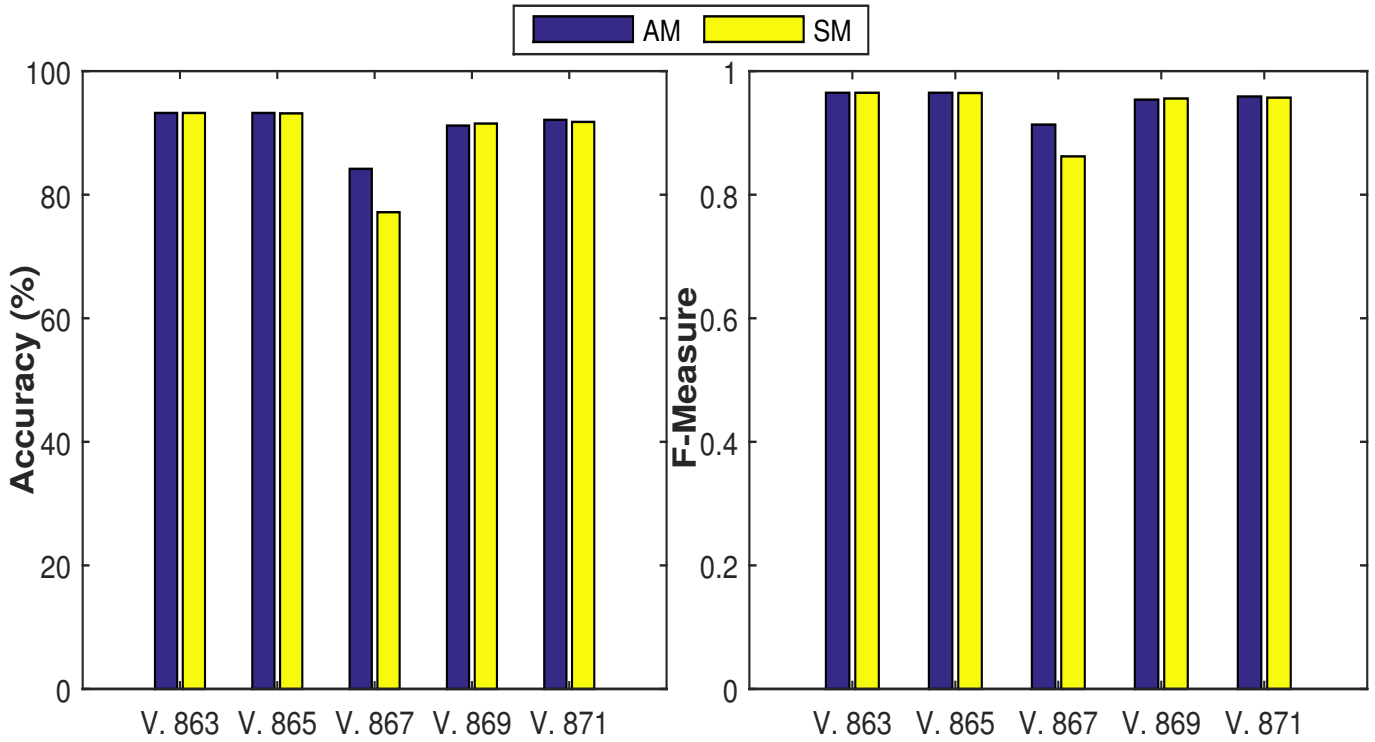


Figure: Validation of prediction models constructed using selected subset of metrics and all metrics (Linear Kernel)

Performance Parameters

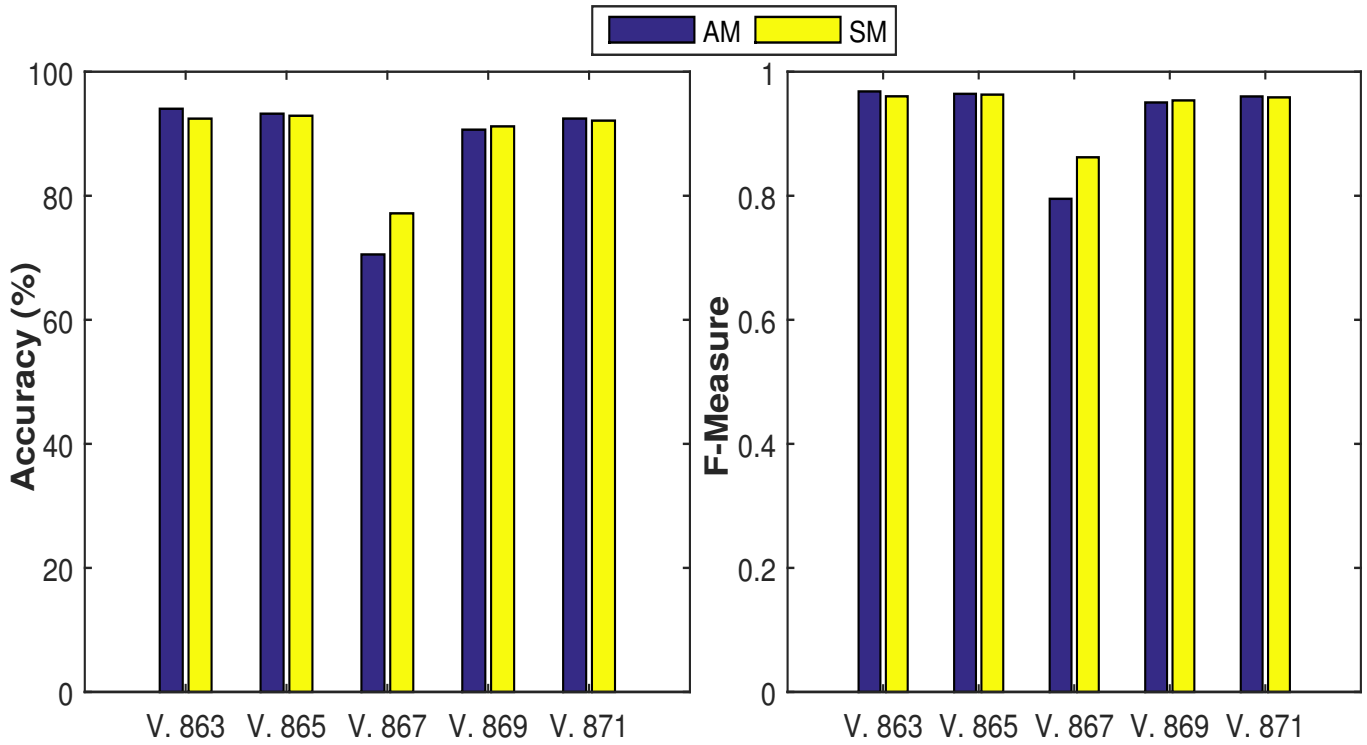


Figure: Results of the validation of prediction models constructed using selected subset of metrics and all metrics (Polynomial Kernel)

Performance Parameters

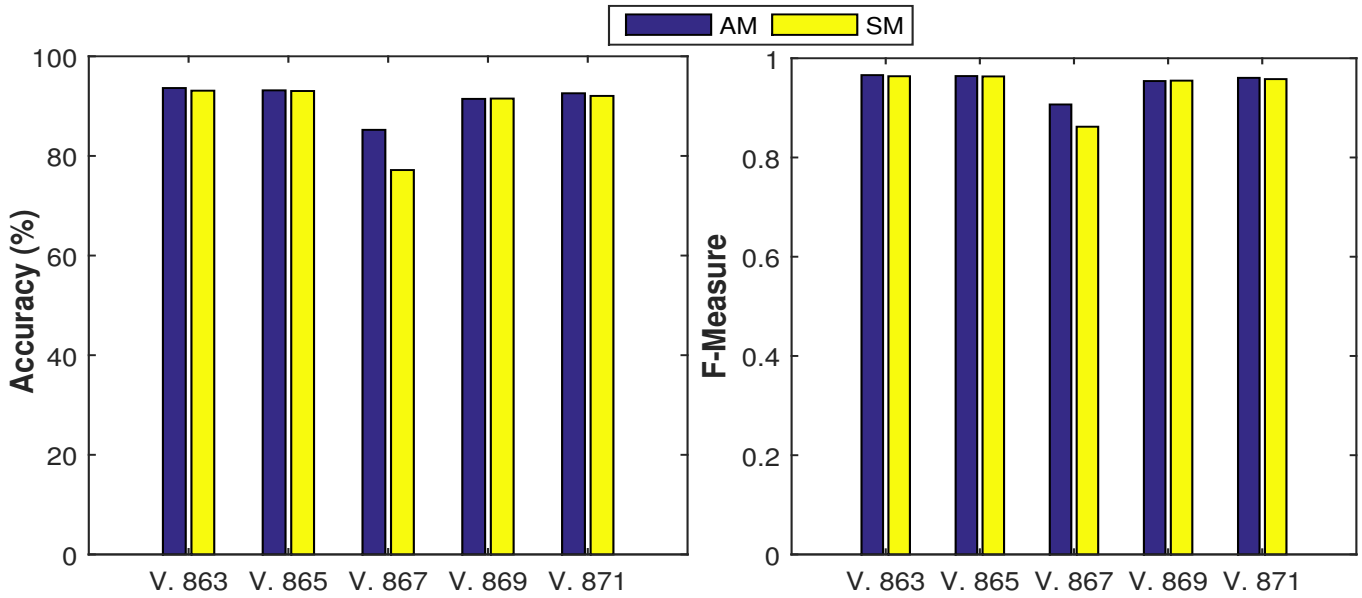


Figure: Results of the validation of prediction models constructed using selected subset of metrics and all metrics (RBF Kernel)

Performance Parameters

1. The y-axis represents the accuracy and f-measure performance evaluation metrics.
2. The two bars represents the performance for the AM and SM metrics respectively.
3. Performance values for different version of eBay web service using LSSVM with linear, polynomial and radial basic function is shown in previous figures.

Observations

Takeaways from Experiments

We observe that change-proneness model developed using selected set of source code metrics passes the desired prediction accuracy and comparable with the models that are build by considering all twenty one metrics.

This results confirm that the ability of these selected source code metrics to predict change-proneness in the eBay web service.

Table of Contents

Table: Feature Selection Techniques

Accuracy						
	P-value		**t-value**		**Mean Difference**	
	AM	SM	AM	SM	AM	SM
AM	1.00	0.80	0.00	0.25	0.00	0.56
SM	0.80	1.00	-0.25	0.00	-0.56	0.00
F-Measure						
	P-value		**t-value**		**Mean Difference**	
AM	1.00	0.99	0.00	0.01	0.00	0.00
SM	0.99	1.00	-0.01	0.00	0.00	0.00

Source Code Metrics Validation

Two sets (one for each performance measure) are used, each with 15 data points (3 classifier * 5 dataset)

We observe that there is no significant difference between these approaches due to the fact that p-value greater than 0.05.

Based on value of mean difference, we observe that the model developed using all metrics resulted in sightly better (only 0.57 % higher accuracy) results compared to models using selected set of metrics.

Table: Classification Methods

Accuracy									
	P-value			**t-value**			**Mean Difference**		
	Lin	Poly	RBF	Lin	Poly	RBF	Lin	Poly	RBF
Lin	1.00	0.56	0.97	0.00	0.59	0.04	0.00	1.71	0.09
Poly	0.56	1.00	0.60	-0.59	0.00	-0.54	-1.71	0.00	-1.62
RBF	0.97	0.60	1.00	-0.04	0.54	0.00	-0.09	1.62	0.00
F-Measure									
	P-value			**t-value**			**Mean Difference**		
Lin	1.00	0.41	0.72	0.00	0.84	0.36	0.00	0.02	0.00
Poly	0.41	1.00	0.59	-0.84	0.00	-0.55	-0.02	0.00	-0.01
RBF	0.72	0.59	1.00	-0.36	0.55	0.00	0.00	0.01	0.00

Kernel Methods

LSSVM method with three different kernel methods are considered to develop a model to predict class change-proneness

We observe that there is no significant difference between these approaches, due to the fact that the p-value is greater than 0.05

However, upon closely inspecting the value of mean difference, LSSVM with linear kernel function yields better results compared to other kernels.

RQ1 - Possibility of Predicting

We infer that it is possible to predict change-proneness of web services (defined using a WSDL document) using source code metrics implementing the web services.

The accuracy of the predictive models are above 80% for all the three types of Kernel and for all the five version of the metrics.

A consistant accuracy of above 80% for both the metrics set (AM and SM) is an evidence of the effectiveness of the proposed approach.

RQ2 - Performance Variation

We conclude that the performance of the LSSVM method varies with the different set of source code metrics.

Selection of classification metrics to develop a model for predicting change-proneness classes is affected by the selection of source code metrics.

AM metrics outperform the SM metric for the linear kernel and RBF kernel.

The AM metric does not dominate SM metric for the polynomial kernel.

Conclusion

It is possible to predict change proneness of WSDL documents and services using source code metrics and kernel based learning techniques.

The model developed using all metrics results in sightly better (only 0.57% higher accuracy) results compared to models on selected set of metrics.

The predictive model developed using LS-SVM linear kernel yields better results compared to other kernels.

Conclusion

The performance of the selected set of source code metrics varies with the different classification methods (such as linear, polynomial and RBF kernel).

Even after removing 85.71% (Average) of the available number of source code metrics, the developed change-proneness prediction models were not adversely affected; in fact, in some of the cases the results were better.

References I

[1] KK Aggarwal, Yogesh Singh, Arvinder Kaur, and Ruchika Malhotra. Application of artificial neural network for predicting maintainability using object-oriented metrics. *Transactions on Engineering, Computing and Technology*, 15:285–289, 2006.

[2] Shyam R Chidamber and Chris F Kemerer. *Towards a metrics suite for Object-Oriented design*, volume 26. ACM, 1991.

[3] José Luis Ordiales Coscia, Marco Crasso, Cristian Mateos, Alejandro Zunino, and Sanjay Misra. Analyzing the evolution of web services using fine-grained changes. In *ICWS*, pages 392–399, 2012.

[4] José Luis Ordiales Coscia, Marco Crasso, Cristian Mateos, Alejandro Zunino, and Sanjay Misra. Predicting web service maintainability via object-oriented metrics: a statistics-based approach. In *Computational Science and Its Applications–ICCSA 2012*, pages 19–39, 2012.

[5] Francisco Curbera, Matthew Duftler, Rania Khalaf, William Nagy, Nirmal Mukhi, and Sanjiva Weerawarana. Unraveling the web services web: an introduction to soap, wsdl, and uddi. *IEEE Internet computing*, 6(2):86, 2002.

References II

[6] Shyamala Doraisamy, Shahram Golzari, Noris Mohd, Md Nasir Sulaiman, and Nur Izura Udzir. A study on feature selection and classification techniques for automatic genre classification of traditional malay music. In *ISMIR*, pages 331–336, 2008.

[7] A Güneş Koru and Hongfang Liu. Identifying and characterizing change-prone classes in two large-scale open-source products. *JSS*, 80(1):63–73, 2007.

[8] Lov Kumar, Santanu Rath, and Ashish Sureka. Predicting quality of service (qos) parameters using extreme learning machines with various kernel methods. In *Workshop on Quantitative Approaches to Software Quality (QuASoQ 2016) co-located to (APSEC 2016)*. CEUR, 2016.

[9] Ruchika Malhotra and Anuradha Chug. Application of group method of data handling model for software maintainability prediction using object oriented systems. *International Journal of System Assurance Engineering and Management*, 5(2):165–173, 2014.

[10] Ruchika Malhotra and Megha Khanna. Investigation of relationship between object-oriented metrics and change proneness. *IJMLC*, 4(4):273–286, 2013.

References III

[11] Cristian Mateos, Marco Crasso, Alejandro Zunino, and Jos Luis Ordiales Coscia. Detecting wsdl bad practices in code–first web services. *International Journal of Web and Grid Services*, 7(4):357–387, 2011.

[12] Eric Newcomer and Greg Lomow. *Understanding SOA with Web services*. Addison-Wesley, 2005.

[13] José Luis Ordiales Coscia, Marco Crasso, Cristian Mateos, and Alejandro Zunino. Web service interface quality through conventional object-oriented metrics. *CLEI Electronic Journal*, 16(1):5–5, 2013.

[14] Daniele Romano. Analyzing the change-proneness of service-oriented systems from an industrial perspective. In *ICSE*, pages 1365–1368, 2013.

[15] Johan AK Suykens, Lukas Lukas, and Joos Vandewalle. Sparse least squares support vector machine classifiers. In *ESANN*, pages 37–42. Citeseer, 2000.

[16] Yuming Zhou and Hareton Leung. Predicting object-oriented software maintainability using multivariate adaptive regression splines. *JMPT*, 80(8):1349–1361, 2007.