Optimality of Naïve Bayes Classifier in Comparison to Other Complex Classifiers

Sandeep Katypally
NC State University
Centennial Campus
Raleigh
skatypa@ncsu.edu

Michael Herzog
NC State University
Centennial Campus
Raleigh
mherzog@ncsu.edu

Tim Menzies
NC State University
Centennial Campus
Raleigh
tjmenzie@ncsu.edu

ABSTRACT

The simple Bayes classifier is known to be optimal when the attributes are independent in the data given the classes. However, Pazani et al [1] suggest that Naïve Bayes performs well in many domains which contain attribute dependencies and they also suggest that this classifier often outperforms more powerful classifiers. We reproduce these suggestions and verify if it is consistent with software datasets. We also check how Naïve Bayes and other classifiers perform when the SMOTE method is performed on the skewed dataset.

In this paper, we compare Naïve Bayes classifier with 9 other classifiers under different conditions and give an analysis of how Naïve Bayes compares against the other classifiers.

CCS Concepts

Computing methodologies → Supervised learning by classification • Computing methodologies → Cross-validation;

Keywords

SMOTE, sklearn,

1. INTRODUCTION

In the field of machine learning, supervised algorithms are those where classification models learn the model from a set of training examples and their corresponding class labels, then outputs a classifier. Now, the classifier takes unlabeled data and assigns it to the class label.

Over the years, many supervised learning algorithms came into existence. But, there has been no one algorithms which could be used for all kinds of datasets. Many studies have been done to produce the best supervised(classification) algorithms, but there is no one algorithm which performs better than every other algorithm as per the literature.

2. DATA

In this paper, we consider the software engineering defect Prediction datasets which are taken from the PROMISE repository. These datasets are used to classify the software components into being defective or non-defective.
The datasets contain the CK metrics which aim at measuring whether a piece of code follows OO principles or not.

Some of the Object-Oriented Design(OOD) attributes are:
• **Weighted Methods for Class (WMC):** The sum of the complexities of each method in a class. If all the method complexities are considered equal and have the value of 1, then WMC equals the number of methods in a class.
• **Depth of Inheritance Tree (DIT):** Number of classes that this class inherits from.
• **Number of Children (NOC):** The count of immediate subclasses of a class.
• **Response for Class (RFC):** The number of elements in the response set of a class. The response set of a class is the number of methods that can potentially be executed in response to a message received by an object of that class.
• **Lack of Cohesion of Methods (LCOM):** For a class C, LCOM is the number of method pairs that have no common references to instance variables minus the number of method pairs that share references to instance variables.
• **Coupling Between Objects (CBO):** For a class C, CBO is the number of classes that are coupled to C.

There are around 20 attributes in the ck metric datasets taken from the PROMISE repository. We will use the following datasets to perform classifier comparison:

- ANT
- CAMEL
- IVY
- JEDIT
- LOG4J
- LUCENE
- VELOCITY
- SYNAPSE
- XALAN
- XERCES

3. NAÏVE BAYES AND OTHER CLASSIFIERS

10 classifiers including Naïve Bayes have been used to learn the binomial CK metric datasets and compare their performance.

3.1 Naïve Bayes

Naïve Bayes(NB) is a probabilistic classifier which takes advantage of features being independent. It uses the Bayes rule to estimate the probability of a vector instance and labels it as a particular class.

$$p(C_k|\mathbf{x}) = \frac{p(C_k)\, p(\mathbf{x}|C_k)}{p(\mathbf{x})}$$

Figure 1. Bayes Rule to find the posterior probability given the prior probabilities.

3.2 Logistic Regression

Logistic regression is a regression model where the dependent variable is categorical. Logistic regression is a specific case of a linear regression. It is used to solve the problem where the dependent variable pass/fail is represented by "1" and "0".

$$F(x) = \frac{1}{1 + e^{-(\beta_0 + \beta_1 x)}}$$

F(x) is is interpreted as the probability of the dependent variable equaling a "success" or "failure"

3.3 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is a method used to find a linear combination of features that characterizes or separates two or more classes of objects or events. LDA is closely related to various methods including ANOVA, regression analysis PCA and factor analysis.

3.4 Quadratic Discriminant Analysis

Quadratic discriminant analysis (QDA) is closely related to linear discriminant analysis(LDA). QDA assumes the measurements from each class are normally distributed. It is used when a linear model will not suffice and two or more classes must be separated by a quadratic surface.

3.5 Multi-Layer Perceptron or Artificial Neural Network

Artificial Neural Network (ANN) is an information processing algorithm that is inspired by the way biological nervous systems process information. An ANN comprises of large number of nodes connected to each other than solve a problem in unison. Each individual node may have a summation function which combines the weighted values of all its inputs together.

3.6 Random Forests

Random forests is an ensemble classification algorithm that operates by constructing a multitude of decision trees. Decision trees are a popular method for various machine learning tasks. Random forests can also be used to rank the importance of variables in a regression or classification problems.

3.7 K-Nearest Neighbors

K nearest neighbors is a simple algorithm that stores all available cases and classifies new cases based on a similarity measure. KNN can be used for both classification and regression predictive problems. However, it is more widely used in classification problems in the industry.

3.8 Support Vector Machine

Support Vector Machine (SVM) is a supervised machine learning algorithm that is used to analyzed data for classification and regression analysis. It is a non-probabilistic binary linear classifier. SVM can be used as a non-linear classifier by using something called a 'kernel trick'

3.9 Decision Tree

Decision tree learning uses a decision tree and makes a predictive model by maping observations about an item to the conclusions about the item's target or label. It is one of the predictive modelling approach used in machine learning and data mining.

3.10 Perceptron

Perceptron is a kind of linear classifier. This classifier allows for online learning, which means it trains training set elements one at a time. It maps its input x to and output value y through a function f(x).

4. DATA PREPROCESSING

We discretize the datasets, where numbers are converted into finite number of bins, using the following discretization schemes:

Equal Frequency Discretization(Nbins):

Divides every attribute data into n bins which have equal number of elements.

Equal Width Discretization (Histogram discretization):

Divides every attribute data into n bins such that range or every bin is equal

These simple discretization schemes are also very effective on performance of the classifier. Dougherty95[33] . The plots shown in the results section are constructed after performing Equal Frequency Discretization. Please look at the appendix to find results when Equal width discretization is performed.

5. EVALUATION METRICS

Classifiers are evaluated using the standard performance measures which are defined as follows:

	p' (Predicted)	n' (Predicted)
p (Actual)	True Positive	False Negative
n (Actual)	False Positive	True Negative

Recall is a fraction of relevant instances that are correctly predicted

$$\text{Recall} = \frac{TP}{TP + FN}$$

Precision is a fraction of correctly predicted instances that are relevant

$$\text{Precision} = \frac{TP}{TP + FP}$$

F Beta Score(beta=1) is the harmonic mean of precision and recall

$$F = \frac{2}{\left(\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}\right)}$$

Accuracy is the fraction of total correct prediction to total number of predictions.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN}$$

6. EXPERIMENT

6.1 Setup

Python with Numpy and Sci-Kit Learn libraries been used to perform the experiment on a Macintosh operating system.

10 Classifiers as mentioned in the previous sections including Naïve Bayes has been used on 10 software datasets. The datasets are taken from the Promise repository[2].

6.2 Preprocessing

Discretization of the datasets has been performed before any further processing. 2 kinds of discretization have been used on the datasets 1) Equal Frequency Discretization 2) Equal Width Discretization.

SMOTE technique has been performed on the training dataset since the datasets are skewed (i.e positive to negative ratio of the labels was much greater or less than 1).

6.3 Results

Naïve Bayes classifier is compared to other classifiers for every dataset. A plot per performance metric i.e. for accuracy, precision, recall, F beta score and run time have been plotted as below.

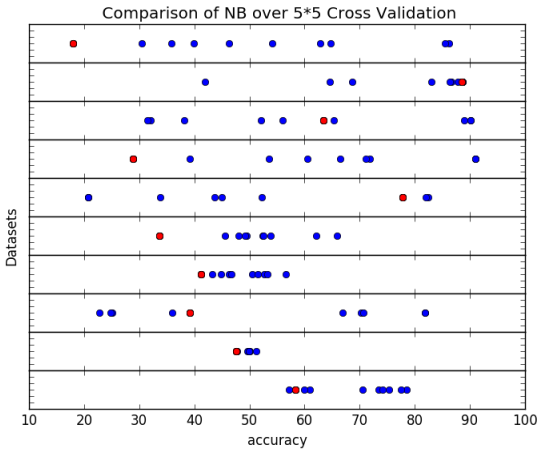

Figure 1. Accuracy comparison of 10 datasets on 10 classifiers 5×5 cross-validation. Red dot represents the Naïve Bayes classifier; blue dots represents other 9 classifiers. SMOTE has been performed on the training dataset only.

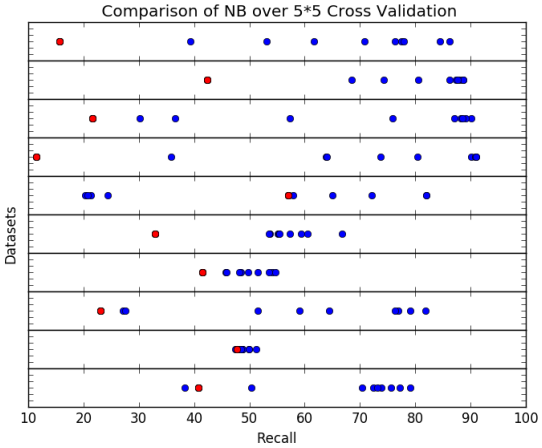

Figure 2. Recall comparison of 10 datasets on 10 classifiers 5×5 cross-validation. Red dot represents the Naïve Bayes classifier; blue dots represents other 9 classifiers. SMOTE has been performed on the training dataset only.

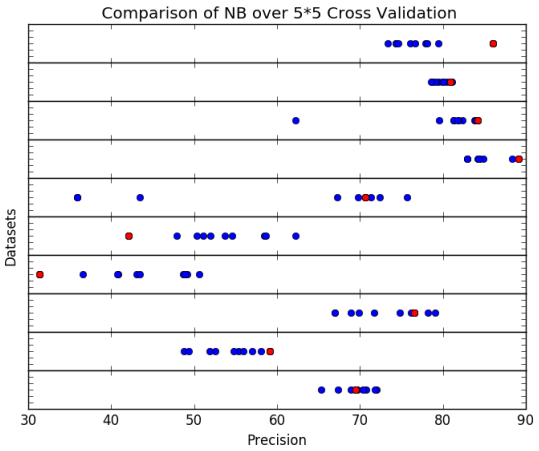

Figure 3. Precision comparison of 10 datasets on 10 classifiers 5×5 cross-validation. Red dot represents the Naïve Bayes classifier; blue dots represent other 9 classifiers.

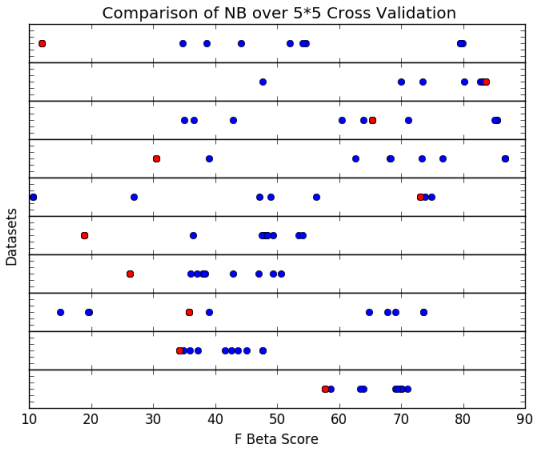

Figure 4 F Beta Score comparison of 10 datasets on 10 classifiers 5×5 cross-validation. Red dot represents the Naïve Bayes classifier; blue dots represent other 9 classifiers.

From the above plots, in most of the datasets accuracy of the NB classifier is worse than most classifiers. Likewise, recall of NB is also worse compared to most other classifiers. However, precision of NB is better than most of the classifiers. F Beta score(Beta=1) is worse for NB classifier. In some cases, F Beta score of NB is the worst compared to other classifiers. The performance of the NB classifier is inconsistent throughout 10 datasets but mostly performing worse than other classifiers.

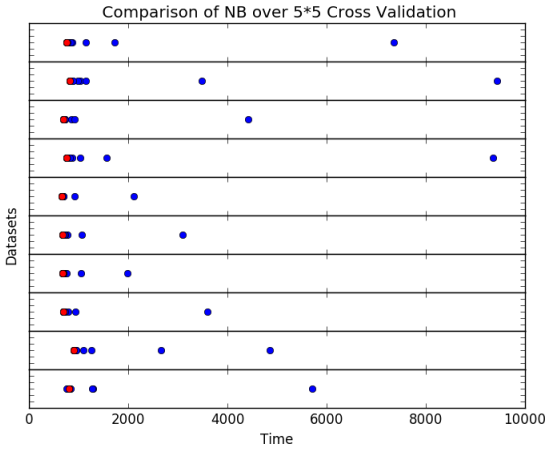

Figure 5. Run-time comparison of 10 datasets on 10 classifiers 5×5 cross-validation. Red dot represents the Naïve Bayes classifier; blue dots represent other 9 classifiers.

From the above run-time graph, NB classifier runs faster than always when compared to other classifiers. This is the only consistency we have seen for NB so far. We also plotted the performance metrics of the classifiers on datasets without applying SMOTE on them so that we could compared the performance of NB and other classifiers due to SMOTE.

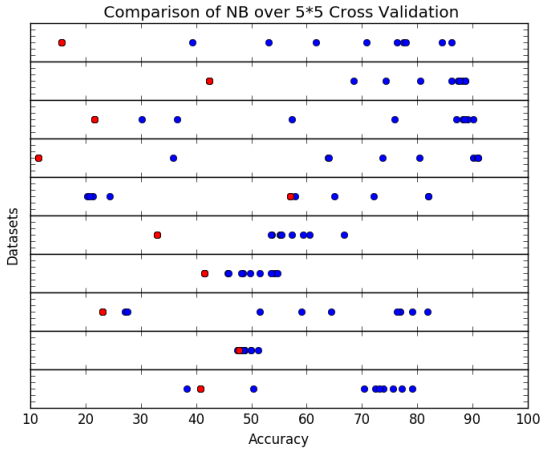

Figure 6 Accuracy comparison of 10 datasets on 10 classifiers 5×5 cross-validation with NO Smote.

Accuracy is much worse for NB compared to when we did SMOTE.

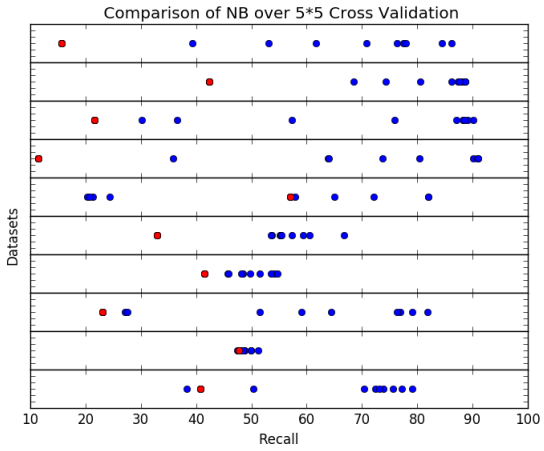

Figure 6 Recall comparison of 10 datasets on 10 classifiers 5×5 cross-validation with NO Smote.

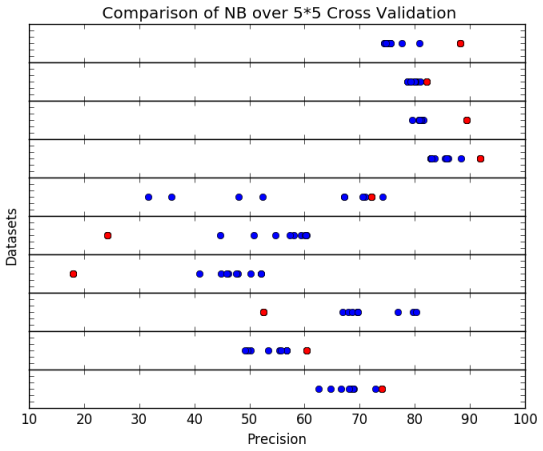

Figure 6 Precision comparison of 10 datasets on 10 classifiers 5×5 cross-validation with NO Smote.

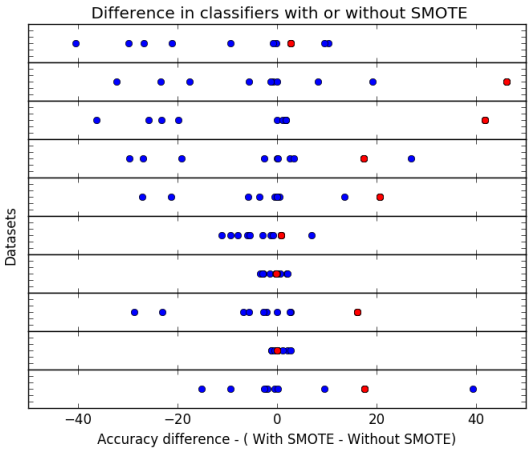

Figure 6 Accuracy difference comparisons of classifier with and without Smote on 10 datasets on 10 classifiers performing 5×5 cross-validation.

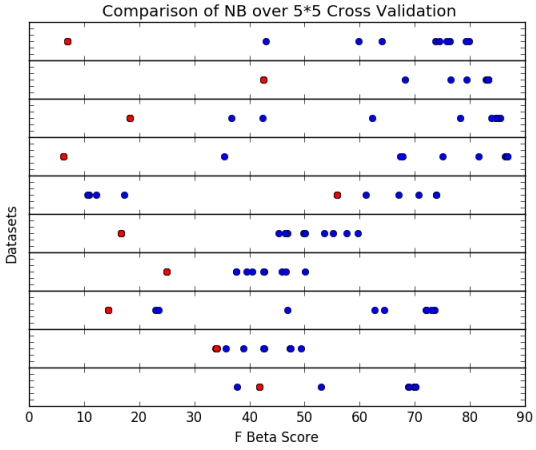

Figure 6 F beta score comparison of 10 datasets on 10 classifiers 5×5 cross-validation with NO Smote.

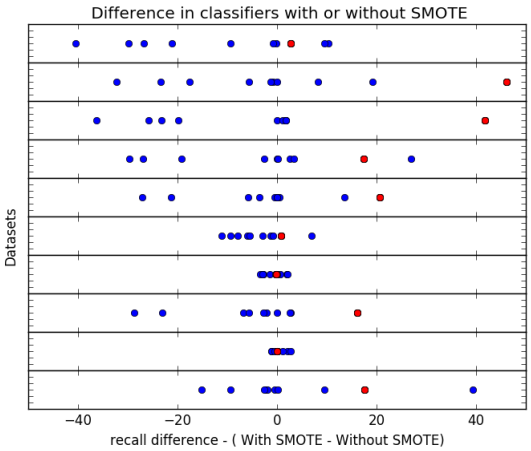

Figure 6 Recall difference comparisons of classifier with and without Smote on 10 datasets on 10 classifiers performing 5×5 cross-validation.

To understand the effect of Smote better, following plots showing the difference in the performance metrics of the classifiers for same datasets with and without Smote have been plotted. Where x-axis = Accuracy (Smote) - Accuracy (NO Smote) and, y-axis = different datasets.

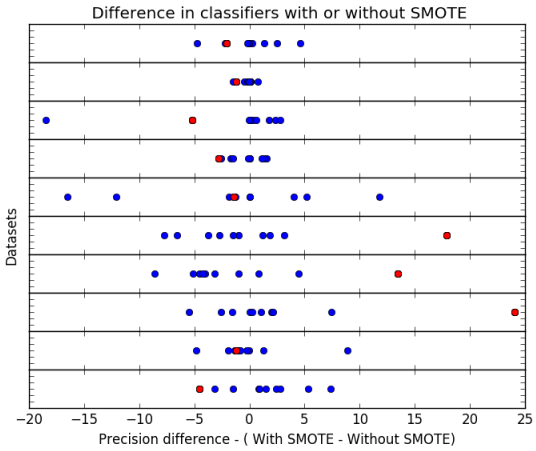

Figure 6 Precision difference comparisons of classifier with and without Smote on 10 datasets on 10 classifiers performing 5×5 cross-validation.

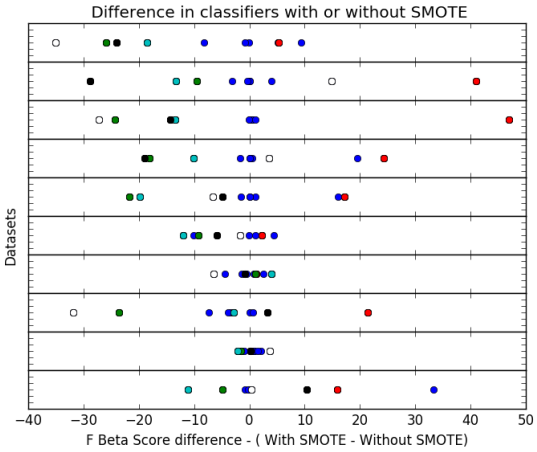

Figure 6 F Beta score difference comparisons of classifier with and without Smote on 10 datasets on 10 classifiers performing 5×5 cross-validation. Red=NB, Light Blue = Random Forests, Green= KNN, **Black**= CART, White=Perceptron, **Blue**=Others

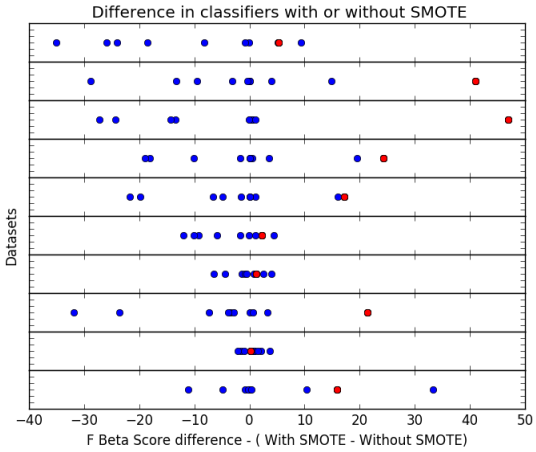

Figure 6 F beta score difference comparisons of classifier with and without Smote on 10 datasets on 10 classifiers performing 5×5 cross-validation.

From the performance difference plots of classifier on with and without Smote, performance of NB with Smote is either equal to and sometimes significantly greater performance of NB without Smote. Therefore, it can be said that Smote improves performance of NB classifier. Hence Smote is a good preprocessing step for Naïve Bayes classifier. Taking this further and coloring some of the other classifiers which are performing worse with Smote.

Performance of all the other classifiers stays the same or deteriorates. Specifically, performance of KNN, CART, Random Forests and Perceptron deteriorate significantly. Remaining classifiers do not improve nor deteriorate significantly. Therefore, Smote is not suggested when applying classifiers like KNN, CART, Random Forests and Perceptron.

7. CONCLUSION

Naive Bayes classifier performs comparatively worse than most other complex classifiers on many of the CK metric datasets.

SMOTE helps improve performance of Naïve Bayes classifier on all CK metric datasets

SMOTE deteriorates the performance of KNN, CART, Random Forests and Perceptron significantly on all CK metric datasets.

SMOTE does not affect other classifiers (Artificial Neural Networks, LDA, QDA, Decision Tree, Support Vector Machine) significantly.

8. FUTURE WORK

Consider more classifiers and find out which classifiers to not use at all and which of more classifiers should not be used with Smote.

9. ACKNOWLEDGMENTS

I thank Dr. Tim Menzies for his encouragement to pursue this project.

10. REFERENCES

[1] Bowman, M., Debray, S. K., and Peterson, L. L. 1993. Reasoning about naming systems. *ACM Trans. Program. Lang. Syst.* 15, 5 (Nov. 1993), 795-825. DOI= http://doi.acm.org/10.1145/161468.16147.

[2] Ding, W. and Marchionini, G. 1997. *A Study on Video Browsing Strategies*. Technical Report. University of Maryland at College Park.

[3] Fröhlich, B. and Plate, J. 2000. The cubic mouse: a new device for three-dimensional input. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (The Hague, The Netherlands, April 01 - 06, 2000). CHI '00. ACM, New York, NY, 526-531. DOI= http://doi.acm.org/10.1145/332040.332491.

[4] Tavel, P. 2007. *Modeling and Simulation Design*. AK Peters Ltd., Natick, MA.

[5] Sannella, M. J. 1994. *Constraint Satisfaction and Debugging for Interactive User Interfaces*. Doctoral Thesis. UMI Order Number: UMI Order No. GAX95-09398., University of Washington.

[6] Forman, G. 2003. An extensive empirical study of feature selection metrics for text classification. *J. Mach. Learn. Res.* 3 (Mar. 2003), 1289-1305.

[7] Brown, L. D., Hua, H., and Gao, C. 2003. A widget framework for augmented interaction in SCAPE. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology* (Vancouver, Canada, November 02 - 05, 2003). UIST '03. ACM, New York, NY, 1-10. DOI= http://doi.acm.org/10.1145/964696.964697.

[8] Yu, Y. T. and Lau, M. F. 2006. A comparison of MC/DC, MUMCUT and several other coverage criteria for logical decisions. *J. Syst. Softw.* 79, 5 (May. 2006), 577-590. DOI= http://dx.doi.org/10.1016/j.jss.2005.05.030.

[9] Spector, A. Z. 1989. Achieving application requirements. In *Distributed Systems*, S. Mullender, Ed. ACM Press Frontier Series. ACM, New York, NY, 19-33. DOI= http://doi.acm.org/10.1145/90417.90738.

Columns on Last Page Should Be Made As Close As Possible to Equal Length