Assessing the Quality and Cleaning of a Software Project Dataset: An Experience Report

Gernot Liebchen
Brunel University, UK
gernot.liebchen@brunel.ac.uk

Bheki Twala
Brunel University, UK
bhekisipho.twala@brunel.ac.uk

Martin Shepperd
Brunel University, UK
martin.shepperd@brunel.ac.uk

Michelle Cartwright
Brunel University, UK
michelle.cartwright@brunel.ac.uk

Abstract

OBJECTIVE - The aim is to report upon an assessment of the impact noise has on the predictive accuracy by comparing noise handling techniques.
METHOD - We describe the process of cleaning a large software management dataset comprising initially of more than 10,000 projects. The data quality is mainly assessed through feedback from the data provider and manual inspection of the data. Three methods of noise correction (polishing, noise elimination and robust algorithms) are compared with each other assessing their accuracy. The noise detection was undertaken by using a regression tree model.
RESULTS - Three noise correction methods are compared and different results in their accuracy where noted.
CONCLUSIONS - The results demonstrated that polishing improves classification accuracy compared to noise elimination and robust algorithms approaches.

Keywords: Software Engineering, Data Quality, Filtering, Polishing, Robust Algorithms

1. INTRODUCTION

This work is motivated by some of the challenges and issues associated with the preparation of a large industrial software engineering dataset which was made available for us to analyse. Specifically, we wished to assess the quality of the data prior to its analysis and, if possible, remove that data we considered to be suspect or low quality. Failure to do so could, at best lead to a waste of research resources, and at worst lead to misleading or inappropriate conclusions being drawn from the analysis.

Problems with the quality of the data can arise for many reasons, particularly when collected from the field. Our dataset was collected over a number of years from many different countries. This can lead to differing interpretations of the data requirements and certainly very different contexts in which the data were collected. To compound matters the data were used by senior management to assess software project progress, a fact that those collecting the data were all too aware of.

Thus this paper describes some of our experiences and ideas about how to go about evaluating the quality of industrial datasets. Clearly this is a commonplace problem so the next section briefly reviews some related work and terminology. This is followed by a description of our dataset and some background information on how it was collected. This is followed by an outline of a technique based upon rule induction and tree pruning to identify outlier cases and features. We then describe how the results apply to our dataset prior to a discussion of the possible implication of these ideas and future work.

2. RELATED WORK

2.1. Issues with Noise and Outliers

Data imperfections pose a problem for researchers investigating real world data. These imperfections can have unwanted impact on the data analysis and might lead to false conclusions about the data. One form of these imperfections are random errors or "noise" which is inevitable in real world data. "Noise" can be caused by different factors for instance spurious correlations (i.e. correlations that are due mostly to the influences of one or more "other" variables), attributes that are not recorded, two examples having the same attribute/value pairs but different classifications, some values of attributes being incorrect because of errors in the data acquisition process or the processing phase, values of attributes being missing, and the classification or class label being wrong (for example, 1 instead of 2) because of some error. Section 4 of this paper indicates some known and suspected sources of "noise" in the provided dataset. More extensive information on sources of "noise" is provided by Manago and Kodratoff [13].

Another problem with datasets are outliers. Outliers can be defined as discordant observations for contaminants which are noted by the data analyst due to their difference to rest of the dataset [11]. Contaminants are data points which where created as part of different distributions from the rest of the data points. These contaminants will not always be discovered by the analyst [1]. However, it must be stressed that outliers are not necessarily problematic and it is not always appropriate to remove them. This is in contrast to noisy observations.

The next problem is also related to unnecessary attributes (which can be caused by noise) which, besides making no contribution to the predictive performance of the learning system, will simultaneously impose an extra computational burden. This situation is generally referred to as overfitting [18, 7, 6], i.e. overfitting to the training example data. For example, if the hypothesis space has many dimensions because of a large number of attributes, meaningless regularity maybe be found in the data that is irrelevant to the true, important, distinguishing features. Overfitting is harmful for several reasons. First, overfitted models are incorrect; they indicate that some variables are related when they are not. Second, overfitted models are difficult to understand due to the unnecessary component that complicates attempts to integrate induced models with existing knowledge derived from other sources, and overfitting avoidance has sometimes been justified solely on the grounds of producing comprehensible models. Finally, overfitted models can have lower accuracy on new data than models that are not overfitted as demonstrated with a variety of domains and systems by Quinlan [15]. In the area of decision tree learning, overfitting is avoided or fixed, to a certain extent, by:

1. Termination of tree growth when further splitting the data does not yield a statistically significant improvement, or by
2. Growing a full tree, then pruning the tree by eliminating nodes.

In practice the latter approach has been more successful.

2.2. Approaches to handle noise and outliers

There are three main approaches of the handling of outliers and noise. First, the imperfections can be ignored and left in the source unchanged, second the imperfections can be removed, or third the imperfections can be corrected [21].

Ignoring the imperfections and leaving the noise in the data set would require a robust algorithm [5] such that overfitting of the data is avoided.

The issue with ignoring noise and outliers is that they can influence findings and therefore result in incorrect conclusions of the data analysis [21]. Excluding the imperfections from the dataset limits the impact of spurious findings but is information inefficient because less data is available for the analysis. This approach works by filtering the data according to criteria and removing the cases which do not adhere with the criteria and was successfully used by John [12], Gamberger et.al. [8], and Rousseeuw and Leroy[17]. Of course this requires that we have an effective method to identify the outlier data. John stresses the importance of a robust algorithm when using the filtering approach. The algorithm also needs to be flexible which would

increase its accuracy [12]. Gamberger et al. state that their elimination algorithm avoids overfitting and that the results of the noise elimination process will not influence the following analysis of the data and that therefore a hypothesis based on this analysis will not be influenced by the noise [8]. In [9] Gamberger et al. compare different noise elimination algorithms and note that a less rigid algorithm might be more precise than one which eliminates a high number of noisy cases. Brodley and Friedl [3] note that it is important to find good algorithms which distinguish between noise and exceptional data cases such that valid data does not get deleted during the noise elimination process. Filtering does despite its drawbacks of being rigid and possibly resulting in the elimination of valid data provide an effective technique for handling noise as it identifies noise and generally avoids overfitting.

Correcting the imperfections has the advantage that more data for the analysis of the dataset will be available. It is based on two phases. The first step is the prediction of the correct outcome of the imperfect data which is followed by the adjustment of the imperfection. This method was used by Teng [19][20][21]. Teng states that the correction or polishing can successfully repair the data to some extend and therefore enhance the quality of the dataset.

The three noise handling methods will be compared in sections 5 and 6.

Two main approaches of outlier detection where identified by Hawkins et al. [10]. The detection can either be distribution based or distance based [10], but both approaches might sometimes overlap. Examples for distance based approaches ar the use of Mahalanobis, Euclidean distance, distance clustering, and visualisation methods [10]. They work by looking at the distance of a data point from the main field of dataset.

Distribution based approaches look at the distribution of the data, and search for misclassifications. Examples for these methods might be the use of neural networks [10], forward search algorithms [4], and robust tree modelling [4][12].

The robustness and accuracy of the three methods for tolerating noise in training data is evaluated using a decision tree classifier approach. Such approach is briefly explained in the following section.

3. DECISION TREES

Decision trees (DTs) are one of the most popular approaches for both classification and regression type predictions. They are generated based on specific rules. A DT is a classifier in a tree structure. A leaf node is the outcome obtained and it is computed with respect to the existing attributes. A decision node is based on an attribute, which branches for each possible outcome for that attribute. DTs can be thought as a sequence of questions, which leads to a final outcome. Each question depends on the previous question, hence, this case leads to a branching in the decision tree. While generating the DT, the main goal is to minimise the average number of questions in each case. This task provides increase in the performance of prediction [14]. One approach to create a DT is to use the entropy, which is a fundamental quantity in information theory. The entropy value determines the level of uncertainty. The degree of uncertainty is related to the success rate of predicting the result. Often the training dataset used for constructing a DT may not be a proper representative of the real-life situation and may contain noise and the DT is said to overfit the training data. To overcome the over-fitting problem DTs use a pruning strategy that minimizes the output variable variance in the validation data by not only selecting a simpler tree than the one obtained when the tree building algorithm stopped, but one that is equally as accurate for predicting or classifying "new" observations. For our experiments we used regression trees which may be considered as a variant of decision trees, designed to approximate real-valued functions instead of being used for classification tasks.

There are several possible criteria for evaluating the DT algorithm; for example, the speed of the learning, speed of the classifier, space requirements and predictive accuracy of the classifier. For the purposes of the paper the latter criterion is used. The predictive accuracy of the DT classifier is determined by the percentage of the test data set that is correctly classified.

4. THE DATASET

The dataset used in this investigation contains software management data collected by a large multinational software house we will call X producing software solutions for different industry sectors. The dataset contains project management data collected by this company's development teams since the beginning of the 1990s. The projects vary immensely in size. The original dataset held values for 10434 cases with 22 attributes. Some of these attributes where administrative (e.g. Project Name, Project ID). In addition there is data on time scales, technology, effort by month and application area. The data include both new development and maintenance projects.

However, there are concerns about the quality of the data. The dataset showed that a large number of values where missing. Especially values for effort and size which was measured in function points where seen as crucial since the aim of the future analysis of the dataset is to analyse factors influencing software productivity. Size and effort are seen as crucial for the calculation of software productivity. The dataset also contains cases which seemed to produce extremely high and extremely low levels of productivity. The data provider did also mention that the data might contain noise which was confirmed by a preliminary analysis of the data which also indicated the existence of outliers.

4.1. The Source

As mentioned above the data was entered by the developers themselves. According to the Metrics Specialist (a senior X contact assisting us with questions about the data and the data entry process) there were differences in the attitude of the development teams towards the data entries. Some of the teams might have been more meticulous than others. Information about the quality of the entered data is not given. The data was meant to be"approved" by an "appropriate" person, but this process cannot be retraced either.

Changes were made on the input tool over the years in order to make it more user friendly, but the impact of these changes is not given. The tool does provide the user with some range checking, but the person "approving" the data has the main responsibility of checking the validity of the data. If this was carried out equally thoroughly is not certain.

According to Metrics Specialist the quality of the data is a concern in X. The input tool used to be connected to a dashboard called "Service Excellent" which was controlled by X's senior management. This resulted in cases where development teams did alter the values of the project data in order to avoid negative attention. The dashboard was disconnected due to doubts regarding the honesty of the data input process and the general impression in X is that the data entry process became more honest. This impression cannot be proven.

5. EXPERIMENTAL SET UP

Our experiments are designed to assess the robustness and accuracy of techniques for handling noise in both training and test data sets and the effect that noise has on predictive accuracy. The CART decision tree builder [2] was used to provide our basic classifier in our experiments. The three noise correction techniques are briefly discussed below:

Robust algorithms: The dataset is taken as is, with the noisy instances left in place. This is accomplished by avoiding overfitting, utilizing the pruning strategy when using decision trees, so that the resulting classifier is not overly specialized to account for the noise. This approach is taken for the CART [2] and C.4.5 [16] systems.

Filtering: On this approach, the data is filtered before being used. Instances that are suspected of being noisy are discarded from the training data (according to certain evaluation criteria), and a hypothesis is then built from the set of remaining instances. This is an approach followed by [12][8][3]. Similar ideas exist in robust regression and outlier detection techniques in statistics [17].

Polishing: This approach is similar to filtering. However, the instances suspected to being noisy are not discarded from the training data. Instead, they are repaired or corrected (or polished) by replacing, say, the

Dataset	Noise Level (%)	Instances with Noise (%)	Attribute Noise (%)	Class Noise (%)
Company X	0	0.0	0.0	0.0
	10	92.6	9.6	8.3
	20	99.3	17.2	11.8
	30	100.0	25.6	17.5
	40	100.0	34.1	25.2

Note: An instance, attribute value, or class value is considered noisy if the respective component differs from that of the original instance.

TABLE 1: Noise characteristics of Company X dataset at various noise levels

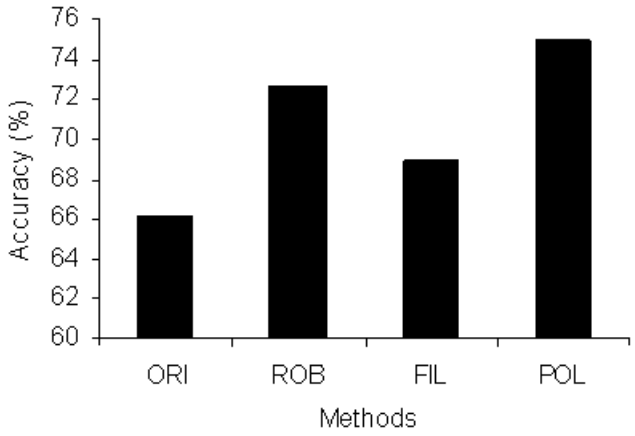

FIGURE 1: Overall means for noise handling methods

corrupted values with more appropriate ones [19][20]. If done correctly, this simplifies the analysis task and produces more complete analysis results.

The technique that shows superior performance shall then be applied to enhance the quality of the data. Since the ultimate test of the quality of induced decision trees is their performance on unknown (unseen) instances, experiments were performed on five different random partitions of the data into 80% training and 20% testing instances. The original dataset consisted of 10434 instances. However, there were instance in the dataset with missing class labels. These instances were removed from the training data, thus, reducing the original size of the dataset to 8911 instances. Each fold is about 1782 instances. The training data was artificially corrupted by introducing labeling or random errors into the attributes using noise levels ranging from 0 to 40%. For nominal attributes, a noise level of x% means that the value of each attribute and the target class is assigned a random value x% of the time, with each alternative value being equally likely to be selected. For a numerical attribute, we select a random value that is between the maximal and the minimal. With this scheme, the actual percentage of noise is always lower than the theoretical noise level, as sometimes the random assignment would pick the original value (especially for nominal attributes). The actual percentages of noise in the various components of the dataset are given in Table 1.

6. EXPERIMENTAL RESULTS

Figure 1 shows the classification accuracy for the Company X data of the decision tree classifier learned from the original noisy training data (ORI), that is, without any noise elimination), a robust algorithm (ROB), a filtering algorithm (FIL), and a polishing (POL). The highest classification accuracy was achieved by POL followed by ROB and FIL, respectively. The worst classification performance is achieved by the classifier when no noise handling technique is utilised before building the model. There also appears to be significant difference between methods at the 5% level of significance.

Artificial Noise Level (%)	0	10	20	30	40
Actual Noise Level (%)	0.0	9.6	17.2	25.6	34.1
Methods:					
Original	73.9	70.8	66.5	61.9	57.3
Robust	79.3	75.2	73.1	70.7	65.0
Filtering	74.8	71.6	69.2	66.7	62.5
Polishing	80.3	77.4	76.1	73.1	67.9

TABLE 2: Classification accuracy - Company X data

In Table 2 we show the accuracy for the Company X data of the DT classifier tested using tested using POL, ROB and FIL. The first row reports the noise rate used to corrupt the data. The actual percentage of corrupted training data is reported in the second row of the table."

When no noise is introduced, there is no significant difference between POL and ROB at the 1% level. However, there is a significant difference between the POL and ROB (individually) and the other remaining two methods. As the noise level increases from 0 to 10%, we begin to observe significant differences between POL and ROB, with ROB achieving higher accuracy rates. There are also significant differences between the other techniques. For noise levels of up to 20%, ROB and POL were able to retain close to their baseline accuracy, which we define to be that obtained for the case of 0% noise and original noisy data. For noise levels of 30% and 40% POL improves accuracy with ROB performing better than FIL at both levels of noise.

7. DISCUSSION AND CONCLUSIONS

The paper presents and empirical evaluation of three noise handling techniques that would enable us to improve the quality of an industrial dataset. The results of an empirical investigation demonstrated that polishing improves classification accuracy compared to noise elimination and robust algorithms approaches. Therefore, identifying and correcting (cleaning) noise from the training data will most likely enhance the accuracy of learned classifiers. Our experiments further showed that as the level of noise increases, the ability of the method to retain the baseline accuracy decreases. The three noise handling techniques were applied on one software engineering project datasets. This work could be extended by considering a more detailed simulation study using much more balanced additional types of datasets required to understand the merits of noise correction. In addition, using as many datasets as possible in our comparative simulation study would enable a more sound generalization of our results. It will also be interesting to examine if cleaning noise from the test data would bring more benefits (in terms of classification accuracy) even if the classifier is learned from a noise corrupted training data.

8. ACKNOWLEDGMENTS

Many thanks to the Metrics Specialist at Company X for the assistance given aiding the understanding the data and data input process of the data set.
The authors also would like to thank Company X for providing the dataset.

REFERENCES

[1] V. Barnett, and T. Lewis, *Outliers in Statistical Data*, 2nd ed., Wiley, New York, 1984.
[2] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*, Wadsworth International Group, Belmont, CA, 1984.
[3] C. E. Brodley and M.A.Friedl, "Identifying mislabelled training data", *Journal of Artificial Intelligence Research*, vol. 11, pp. 131-167, 1999.
[4] R. Chambers, A. Hentges and X. Zhao, "Robust automatic methods for outlier and error detection", emphJournal of the Royal Statistical Society Series A, vol 167, 2004,
[5] P. Clark and T. Niblett, "The CN2 Induction Algorithm", *Machine Learning* ,vol. 3,pp. 261-283,1989.
[6] P. R. Cohen, D. Jensen, "Overfitting Explained", *Preliminary Papers of the Sixth International Workshop on Artificial Intelligence and Statistics*, pp. 115–122, 1997.

[7] R.S. Forsyth, D. D. Clarke, and R. L. Wright, "Overfitting revisited: An information-theoretic approach to simplifying discrimination trees", *Journal of Experimental and Theoretical Artificial Intelligence*, vol. 6, pp. 289-302, 1994.

[8] D. Gamberger, N. Lavrac, and S. Dzeroski, "Noise Elimination in Inductive Concept Learning: A Case Study in Medical Diagnosis", *Proceedings of the 17th International Workshop on Algorithmic Learning Theory*, pp. 199-212, 1996.

[9] D. Gamberger, R Boskovic, N. Lavrac, "Experiments with Noise Filtering in a Medical Domain", *Proceedings of the 16th International Conference on Machine Learning*, pp.143-151, 1999.

[10] S. Hawkins, H. He, G. J. Williams, R. A. Baxter, "Outlier Detection Using Replicator Neural Networks", *DaWaK 2000: Proceedings of the 4th International Conference on Data Warehousing and Knowledge Discovery*, pp. 170-180, 2002.

[11] B. Iglewicz, "Robust scale estimators and confidence intervals for location", *Understanding Robust and Exploratory Data Analysis*, pp. 405431, 1983.

[12] G. H. John, "Robust decision trees: Removing outliers from databases", *Proceedings of the 1st International Conference on Knowledge Discovery and Data Mining*, pp. 174-179, 1995.

[13] M. V. Manago, and Y. Kodratoff, "Noise and Knowledge Acquisition", *The Tenth International Joint Conference on Artificial Intelligence*, vol. 1, pp. 348-349, 1987.

[14] T. M. Mitchell, *Machine Learning*, McGrawHill, 1997.

[15] J.R. Quinlan, "Simplifying decision trees", *International Journal of Man-Machine Studies* , vol. 27, pp. 221–234, Academic Press Ltd., London, UK, 1987.

[16] J. R. Quinlan, *C4.5: Programs For Machine Learning*, Morgan Kaufmann, Los Altos, 1993.

[17] P. J. Rousseeuw and A. M. Leroy, *Robust regression and outlier detection*, John Wiley & Sons, Inc. ,New York, NY, USA, 1987.

[18] C. Schaffer, "Overfitting avoidance as bias", *Machine Learning*, vol. 10, pp. 153-178, 1993.

[19] C. M. Teng, "Correcting noisy data", *Proceedings of the 16th International Conference on Machine Learning*, 239-248, 1999.

[20] C. M. Teng, "Evaluating noise correction", *Proceedings of the 6th Pacific Rim International Conference on Artificial Intelligence*, Springer-Verlag, 2000.

[21] C. M. Teng, "Combining Noise Correction with Feature Selection", *Data Warehousing and Knowledge Discovery*, pp.340-349, 2003.