International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 10 Issue 5 ǁ 2022 ǁ PP. 11-15

Need of Normalization Techniques in Defect Prediction

Amitava Bondyopadhyay, Dr. Abhoy Chand Mandal

Date of Submission: 07-05-2022 Date of acceptance: 22-05-2022

I. Introduction

The main purpose of software development process is to develop high-quality software efficiently. In recent years the demand for software quality has increased quickly. In recent decades the production of large software projects are challenging , costly and time consuming. In order to minimize cost and enhancing the overall efficiency of the testing process , measuring software defects at early stage is particularly significant. So if we estimate at priori the probable faultiness of software , then it could end up giving help on software development activities planning, controlling and executing. Hence our aim should be to discover a low cost method that can be achieved from learning from earlier error to prevent future one. Nowadays quite a few data sets exists which could be mined to find useful facts about defects. From the beginning data mining techniques are applied in constructing software fault prediction for improving the software quality. So we need to identify high risk modules(having high number of faults) at the earliest which can be helpful in quality enhancement effort.

A software defect [16] is an fault, flaw, bug, mistake, breakdown, or fault in a computer program or system that may make an erroneous outcome, or precludes the software from behaving as wished **.**Any software development team for all time want to produce a quality software with least amount of defects. To boost the software quality, high risk modules from the software development should be take out at the earliest. Software defects constantly invite cost in terms of time and quality **.** To identify and correct defects is one of the major research areas nowadays. It is not possible to eliminate each and every defect in one software but it can be minimized and their adverse effects can be reduced.

One of the most challenging aspect while maintaining the quality of a software is the prediction of defective modules in the early stages of software development. Since the cost of correcting a defective software increasing exponentially at the later stage of software development, so we need to identify defects at the earliest. One of the major research area that remains active in last two decades is predicting defective modules. Software defect detection technique mainly depends on the extraction of software metrics from historical data. The defect prediction activity is needed to increase the effectiveness of quality assurance process. One of the promising way to predict the software defect at the early part of SDLC is machine learning. It can be used to detect the hidden pattern in historical software data.

II. Causes of software failure

By a software system we mean a product that helps business domain. But in today's world , in spite of having a lot of experiences , software failure has become a regular feature. Those failure occur due to fault happened in SDLC or due to lack of requirement specification by the customer.

III. Software defect predictor

By defect predictor we mean a tool or method that helps testing in SDLC. As we know that half the cost of software development is in testing, hence it requires a lot of time. So it is important for us to predict where the defect might exists before we start testing. Those defect predictors are used to allocate resources, make ordering of modules, inspect all code etc. They are also useful to know what modules requires rework, so that the development team have the time to rework prior to delivery.

3.1 Application of Machine Learning in Software Defect Prediction

Machine learning is one of the fastest growing field nowadays. It produces various learning algorithms for different application. Among those some of the learning algorithms are beneficial to software engineering. More ever, different machine learning researchers are proposing different algorithm for software fault prediction . From that, we can classify high quality defect models into those that are based on classification, clustering and ensemble methods.

3.2 Machine Learning in Software Defect Prediction

The field of machine learning has been growing rapidly, producing a variety of learning algorithms for different applications. As we discussed one of the application of machine learning is software engineering. The ultimate value of those algorithms is to a great extent judged by their success in solving real-world problems. Therefore,algorithm reproduction and application to new tasks are crucial to the progress of the field. However, various machine learning researchers currently publish for software fault prediction model development. Now, we categorizing successful software defect model into three that are based on classification, clustering and ensemble methods.

IV. Normalization in Data mining

Normalization is the process of scaling an attribute's data such that it falls within a narrower range, like -1.0 to 1.0 or 0.0 to 1.0. It is beneficial for classification algorithms in general. Normalization is typically necessary when dealing with characteristics on various scales; otherwise, it may dilute the efficacy of an equally significant attribute on a lower scale due to other attributes having values on a greater scale. In other words, when numerous characteristics exist but their values are on various scales, this might result in inadequate data models when doing data mining activities. As a result, they are normalized to put all of the characteristics on the same scale.

The data normalization (also referred to as data pre-processing) is a basic element of data mining. Normalization is used to scale the data of an attribute so that it falls in a smaller range, such as -1.0 to 1.0 or 0.0 to 1.0. It is generally useful for classification algorithms of data mining technique.

4.1 Different types of Normalization

Normalization is a procedure that should be followed for each database you create. Normal Forms refers to the act of taking database architecture and applying a set of formal criteria and rules to it. The normalization process is classified as follows: First Normal Form (1 NF), Second Normal Form (2 NF), Third Normal Form (3 NF), Boyce Cod Normal Form or Fourth Normal Form (BCNF or 4 NF), Fifth Normal Form (5 NF), and Sixth Normal Form (6 NF) (6 NF).

Researchers are increasingly relying on data to learn more about their customers. Thus, data analysts have a bigger responsibility to explore and analyze large blocks of raw data and glean meaningful customer trends and patterns out of it. This is known as data mining. Data analysts use data mining techniques, advanced statistical analysis, and data visualization technologies to gain new insights.

One of the common challenges is that, usually, databases contain attributes of different units, range, and scales. Applying algorithms to such drastically ranging data may not deliver accurate results. This calls for data normalization in data mining.

4.2 Why is Normalization in Data Mining Needed?

Data normalization is mainly needed to minimize or exclude duplicate data. Duplicity in data is a critical issue. This is because it is increasingly problematic to store data in relational databases, keeping identical data in more than one place. Normalization in data mining is a beneficial procedure as it allows achieving certain advantages as mentioned below:

- It is a lot easier to apply data mining algorithms on a set of normalized data.
- The results of data mining algorithms applied to a set of normalized data are more accurate and effective.
- Once the data is normalized, the extraction of data from databases becomes a lot faster.
- More specific data analyzing methods can be applied to normalized data.

4.3 Popular Techniques for Data Normalization in Data Mining

There are three popular methods to carry out **normalization in data mining**. They include:

4.3.1 Min Max Normalization

One of the most prevalent methods for normalizing data is min-max Normalization. For each feature, the minimum value is converted to a 0, the highest value is converted to a 1, and all other values are converted to a decimal between 0 and 1. For example, if the minimum value of a feature was 20 and the highest value was 40, 30 would be converted to about 0.5 since it is halfway between 20 and 40. One significant drawback of min-max Normalization is that it does not handle outliers well. For example, if you have 99 values ranging from 0 to 40, and one of them is 100, all 99 values will be converted to values ranging from 0 to 0.4.

4.3.2 Decimal Scaling Normalization

Decimal scaling is another technique for **normalization in data mining**. It functions by converting a number to a decimal point.

4.3.3 Z-Score Normalization

Z-Score value is to understand how far the data point is from the mean. Technically, it measures the standard deviations below or above the mean. It ranges from -3 standard deviation up to +3 standard deviation. Z-score **normalization in data mining** is useful for those kinds of data analysis wherein there is a need to compare a value with respect to a mean(average) value, such as results from tests or surveys.

1. Need for a Normalized data in Software Defect Prediction using NASA Software Defect Data(MDP) Sets

We always want to make good quality software efficiently. But as the day passes making of quality of software become difficult. Especially if the project is a bigger one. If we want to increase the quality of our software system we have to increase the efficiency of software testing .So we need to find defects from the software modules at the beginning. An early estimation of fault will give much help on software building process. Always we should want a low cost mechanism that learns from previous error to prevent future one. In recent research, we would be able to find a few data sets that can be mined to find useful facts about the defects. Hence the aim should be to identify high risk modules.

So a good deal of interest is there for the machine learning techniques to use prediction systems to determine whether a software module is faulty or not. In this regard NASA datasets is used extensively. With so much research we need to group together the individual research in to a coherent body of knowledge. However, there is a significant question about the quality of software defect data sets that have been made publicly available and widely used by researchers. The main cause for worry is about the data integrity, redundancy and inconsistencies among various versions of the NASA data sets in circulation.

1.1 **Study**

Since Machine learning is a data-driven study and so here data sets are archived and shared among researchers. For example the Promise Data Repository has been beneficial in making software engineering data sets publicly accessible. As of now there are ninety six software defect data sets available. Most of them(13 out of the 14 data sets are given by NASA and is available in NASA Metrics Data Program (MDP) website.

Table 1 : Two versions of the NASA defect data sets

Data Set-	Cases-		Features-	
	MDP-	Promise-	MDP-	Promise-
CM1-	505	498	43	22
JM1-	10878	10885	24	22
KC1-	2107	2109	27	22
KC2-	n.a	522	n.a	22
KC3-	458	458	43	40
KC4-	125	n.a	43	n.a
MC1-	9466	9466	42	39
MC2-	161	161	43	40
MW1-	403	403	43	38
PC1-	1107	1109	43	22
PC2-	5589	5589	43	37
PC3-	1563	1563	43	38
PC4-	1458	1458	43	38
PC5-	17186	17186	42	39

Next we have to think between the various types of data quality difficulty that might exist.

TABLE 2: NASA MDP DATA QUALITY ANALYSIS BY FEATURES

	A		B		C		D		E		F	
Data Set	Identical features		Constant features		Features withmissing values		Features with conflicting values		Features with implausible values		Total problem features	
	MDP	Prom	MDP	Prom	MDP	Prom	MDP	Prom	MDP	Prom	MDP	Prom
CM1	2	0	3	0	1	0	2	14	0	6	6	15
JM1	0	0	0	0	0	5	9	15	0	6	9	16
KC1	0	0	0	0	0	0	4	15	0	6	4	16
KC2	n.a.	0	n.a.	0	n.a	0	n.a	14	n.a	6	n.a.	15
KC3	0	0	1	0	.1	0	.0	0	.1	1	3	1
KC4	27	n.a	26	n.a	0	n.a	3	n.a	0	n.a	30	n.a.
MC1	0	.0	1	.0	0	.0	3	.3	1	.1	5	4
MC2	0	0	1	0	1	0	0	0	0	0	2	0
MW1	2	0	3	0	1	0	0	0	0	0	4	0
PC1	2	0	3	0	1	0	4	14	1	6	8	15
PC2	3	0	4	0	1	0	2	2	1	1	8	3
PC3	2	0	3	0	1	0	2	2	1	1	7	3
PC4	2	0	3	0	0	0	7	7	1	1	11	8
PC5	0	0	1	0	0	0	3	3	1	1	5	4

One of the major problem that we have seen is the problem of duplicate cases in data sets .All data sets have the problem of redundancy. More attention should be paid to this problem since the similar cases may be used both for training and testing.

5.2 NASA defect data set has a problem of duplicity, dependency:

The main cause for worry is about the data integrity, redundancy and inconsistencies among various versions of the NASA data sets in circulation. One of the major problem that we have seen is the problem of duplicate cases in data sets .All data sets have the problem of redundancy. More attention should be paid to this problem since the similar cases may be used both for training and testing. So the need of the hour is to make that data set normalized before we use them for research purpose .

5.3 Need for data Normalization in Data set for obtaining good result

Basic element of data mining is data normalization. It is used to convert the source data into another format that allows processing efficiently. The major aim of data normalization is to minimize or remove redundancy or duplicate values from the data set. Thus, normalization in data mining is like pre-processing and preparing the data for analysis. Hence the novel normalization technique, proposed in our work will be helpful in achieving that.

5.4 Conclusion

The NASA data set may have redundancy, inconsistency due to various reasons. Hence in order to get good result while using that data for research purpose, we need to normalize that. Hence a good normalization technique is needed.

References

[1]. K. O. Elish and M. O. Elish, ―Predicting defect-prone software modules using support vector machines,‖ J. Syst. Softw., vol. 81, no. 5, pp. 649– 660, 2008.

[2]. I. Gondra, Applying machine learning to software fault-proneness prediction,‖ J. Syst. Softw., vol. 81, no. 2, pp. 186–195, 2008.

[3]. J. Zheng, Cost-sensitive boosting neural networks for software defect prediction,‖ Expert Syst. Appl., vol. 37, no. 6, pp. 4537–4543, 201.

[4]. NASA Software Defect Datasets [Online]. Available: https://nasa softwaredefectdatasets.wikispaces.com. [Accessed: 01-April-2019].

[5]. NASA Defect Dataset.‖ [Online]. Available: https://github.com/klainfo/NASADefectDataset. [Accessed: 01-April-2019].

[6]. Mudhu Sudhan Chakraborty, Amitava Bondyopadhyay and Tapas Kumar Ghosh,Towards more efficient 3NF determination using reduced functional dependency sets, Advances and Applications in Mathematical Sciences,Mili Publication,2021
[7]. Ms. Puneet Jai Kaur, Ms. Pallavi, Data Mining Techniques for Software Defect Prediction, International Journal of Software and Web Sciences (IJSWS)
[8]. Wahidah Husain, Pey Ven Low, Lee Koon Ng, Zhen Li Ong, Application of Data Mining Techniques for Improving Software Engineering, ICIT 2011 The 5th International Conference on Information Technology
[9]. K.B.S Sastry, Dr.B.V.Subba Rao, Dr K.V.Sambasiva Rao, Software Defect Prediction from Historical Data, International Journal of Advanced Research in Computer Science and Software Engineering.
[10]. Tao Xie, Suresh Thummalapenta, David Lo, and Chao Liu. Data mining for software engineering. *Computer*, 42(8):55–62, 2009.
[11]. Qinbao Song, Zihan Jia, Martin Shepperd, Shi Ying, and Jin Liu. A general software defect-proneness prediction framework. *Software Engineering, IEEE Transactions on*, 37(3):356–370, 2011.
[12]. Ma Baojun, Karel Dejaeger, Jan Vanthienen, and Bart Baesens. Software defect prediction based on association rule classification. *Available at SSRN 1785381*, 2011.
[13]. S Bibi, G Tsoumakas, I Stamelos, and I Vlahavas. Software defect prediction using regression via classification. In *IEEE International Conference on*, pages 330–336, 2006.
[14]. Tim Menzies, Jeremy Greenwald, and Art Frank. Data mining static code attributes to learn defect predictors. *Software Engineering, IEEE Transactions on*, 33(1):2–13, 2007.
[15]. Iker Gondra. Applying machine learning to software fault-proneness prediction. *Journal of Systems and Software*, 81(2):186–195, 2008.
[16]. Ata¸c Deniz Oral and Ay¸se Ba¸sar Bener. Defect prediction for embedded software. In *Computer and information sciences, 2007. iscis 2007. 22nd international symposium on*, pages 1–6. IEEE, 2007.