Mathematical Model for Detecting Outliers in the Two-Dimensional Data of Software Metrics RFC and CBO from Applications in Java

Sergiy Prykhodko[1,2], Lidiia Makarova[1,*], Liudmyla Latanska[1] and Maksym Bryzghalov[1,*]

[1] *Admiral Makarov National University of Shipbuilding, Heroes of Ukraine Ave., 9, Mykolaiv, 54007, Ukraine*
[2] *Odesa Polytechnic National University, Shevchenko Ave., 1, Odesa, 65044, Ukraine*

Abstract

This paper presents a mathematical model in the form of a transformed prediction ellipse for detecting outliers in two-dimensional data of the software metrics RFC (response for a class) and CBO (coupling between object classes) from applications in Java. At present, when the data distribution follows a normal law, it is possible to apply a prediction ellipse to identify outliers. However, for data whose distribution significantly deviates from normality, the use of such a prediction ellipse becomes statistically invalid. In such cases, it is necessary to first normalize the data, construct a prediction ellipse for normalized metrics, and subsequently apply the inverse transformation to obtain a transformed prediction ellipse for the initial data. The dataset used in this study consists of RFC and CBO metrics collected from various open-source Java projects covering different functional areas, architectural styles, and development practices. This diversity ensures the applicability of the model to a wide range of real-world Java projects. Mardia's normality test shows that the distribution of these metrics deviates from multivariate normality. Consequently, it is essential to apply the normalization procedure described above. For this purpose, we employ a bivariate Box−Cox transformation, which enables scale correction and distributional alignment, facilitating the construction of a mathematically valid transformed prediction ellipse. The study aims to analyze Java applications by building a mathematical model capable of identifying outliers in the metric space and characterizing the typical range of variation in the examined applications. This ellipse is defined by a statistical boundary based on the F-distribution, providing a formal confidence region for typical metrics. Outliers are determined by calculating the Mahalanobis distance from the ellipse center and comparing it to a threshold value. The resulting model allows for formal outlier detection and supports visual analysis of typical and atypical metric behavior, aiding in better interpretation of structural anomalies. The practical use of the constructed model was verified on projects that were not involved in its development. Overall, this approach combines normalization, reliable outlier detection, and the construction of prediction boundaries to distinguish between normal and anomalous behavior of the RFC and CBO metrics.

Keywords

Mathematical model, outlier detection, software metric, Java application, prediction ellipse, Box-Cox transformation, Mahalanobis distance[1]

1. Introduction

Statistical analysis of multivariate data plays an important role in many areas, including empirical software engineering [1, 2]. One of the most important tasks of statistical analysis is to detect outliers in data [3, 4]. There are several models for detecting and removing outliers in the data. One of the known models used in statistical analysis is the prediction ellipsoid, in the particular case of two variables - the prediction ellipse. However, there is a problem: this model works only in the case of normal data distribution.

ICST-2025: Information Control Systems & Technologies, September 24-26, 2025, Odesa, Ukraine.

* Corresponding author.
sergiy.prykhodko@nuos.edu.ua (Sergiy Prykhodko); lidiia.makarova@nuos.edu.ua (Lidiia Makarova); liudmyla.latanska@nuos.edu.ua (Liudmyla Latanska); 201166@nuos.edu.ua (Maksym Bryzghalov)
0000-0002-2325-018X (S. Prykhodko); 0000-0003-2903-3001 (L. Makarova); 0000-0001-6473-7624 (L. Latanska); 0009-0007-4728-3334 (M. Bryzghalov)

Well-known software metrics, RFC and CBO, were proposed by Chidamber and Kemerer [5], have been extensively validated as correlating with fault-proneness and change-proneness across numerous software systems [6, 7]. In Booch's definition of object-oriented design (OOD), they are used today to solve different problems, including software quality [8-10]. Despite their utility, distributions in real-world Java systems often deviate substantially from multivariate normality.

Empirical software engineering studies apply various methods of multivariate statistical analysis and Mathematical Modelling. Assuring the validity of such methods and corresponding results is challenging and critical [11]. As it is known [12], many methods of multivariate statistical analysis are based on the assumption that the data is normally distributed. Also, we know [13] that if the data are not normally distributed, it is misleading to draw conclusions based on the normal distribution.

It is known, the approach in multivariate statistics based on the multivariate normalizing transformations, for instance, the use of the Mahalanobis distance for normalized data to detect multivariate outliers in high-dimensional non-Gaussian data [14, 15].

Importantly, Mahalanobis distance allows modeling of ellipsoidal confidence regions, named prediction ellipsoid (or prediction ellipses in the bivariate case), which capture acceptable ranges of software metrics and detect outliers. This integrated statistical approach not only supports anomaly detection but also contributes to a better understanding of software structure in the early development stages of Java applications.

2. Review of the literature

The use of the object-oriented software metrics CBO and RFC has been well established in software engineering. These metrics are widely recognized for their ability to capture the complexity and interdependence of object-oriented designs, which directly influence maintainability, fault proneness, and project effort estimation [10, 16, 17]. All this also applies to Java applications.

Early predominantly utilized models, in which the acceptable ranges of CBO and RFC metrics values were defined separately for each metric, without taking into account their mutual influence [18, 19]. In addition, the two-dimensional data of these metrics are not normally distributed. This model limits the detection of anomalous values of software metrics. To overcome these limits, further research has begun applying multivariate statistical techniques, including normalizing transformations and prediction ellipses, to model the joint behavior of CBO and RFC metrics. The prediction ellipse constructed given covariance matrix provides geometric boundaries, which ensure accurate outlier detection and removal.

The prediction ellipsoid is used in various statistical methods for multivariate data analysis, such as multivariate outlier detection [20] and solving optimization problems [11, 21]. However, its application is limited by the assumption that the data follow a multivariate normal distribution, which is rarely the case. As a result, transformed prediction ellipsoids are used for multidimensional non-Gaussian data.

In [22], the use of the squared Mahalanobis distance for outlier detection in multivariate non-Gaussian data is discussed, based on applying univariate and multivariate normalizing transformations.

Developing the presented technique, a technique for building transformed prediction ellipsoids based on normalizing transformations for multivariate non-Gaussian data is proposed in [23]. The transformed prediction ellipsoid gives the same results as the squared Mahalanobis distance, but using the transformed prediction ellipsoid is more visually apparent.

3. Formulation of the problem

Traditional approaches to outlier detection in software metrics often rely on the assumption of multivariate normality, which allows the use of statistical methods such as the construction of the prediction ellipse. However, in practice, software metrics data collected from real-world Java applications such as RFC and CBO frequently deviate from this assumption. As shown by Mardia's multivariate normality test, the joint distribution of RFC and CBO metrics, when normalized relative to the number of classes (NCL), does not conform to the normal law. This deviation invalidates the direct application of classical predictive ellipses for the initial data.

To address this, we pose the problem of developing a mathematically grounded technique for detecting outliers in heterogeneous, non-normally distributed initial data. The proposed solution involves normalizing metrics through a bivariate Box–Cox normalization, which aligns the data distribution with the assumptions required for the construction of a prediction ellipse. Based on normalized metrics, a prediction ellipse is constructed using statistical boundaries derived from the F-distribution and evaluated using Mahalanobis distance.

The inverse transformation is then applied to the ellipse, resulting in the construction of a transformed prediction ellipse for the initial data. This transformed prediction ellipse defines a region, enabling the identification of outliers in Java applications.

4. Objectives of the study

The study aims to construct a mathematical model in the form of a transformed prediction ellipse based on the object-oriented software metrics RFC and CBO of Java applications. The model applies a bivariate Box–Cox transformation to normalize software metrics, enabling statistically valid outlier detection through F-distribution-based thresholds and Mahalanobis distance. The study further aims to assess the model's effectiveness in identifying outliers in Java applications.

The object of study is the process of building a mathematical model in the form of a transformed prediction ellipse for detecting outliers in two-dimensional data of the software metrics RFC and CBO from applications in Java.

The study's subject is a mathematical model in the form of a transformed prediction ellipse for detecting outliers in two-dimensional data of the software metrics RFC and CBO from applications in Java.

5. Materials and research methods

To achieve the aim of this study, it is necessary to analyze the existing mathematical models for finding outliers in Java applications. The focus is placed on quantitative values of RFC and CBO metrics collected from a diverse sample of open-source Java applications representing a variety of software types and architectural styles.

The study justifies the need for constructing a transformed prediction model, given that the distribution of these object-oriented metrics deviates from multivariate normality, as confirmed by Mardia's test. This deviation renders conventional linear models and unadjusted ellipses statistically invalid for robust anomaly detection.

The bivariate Box-Cox transformation was chosen to normalize multivariate data:

$$Z = \begin{cases} \frac{X^{\lambda}-1}{\lambda}, if\ \lambda \neq 0; \\ \log(X)\,, if\ \lambda = 0, \end{cases} \tag{1}$$

where X takes the values of according columns of data RFC/NCL and CBO/NCL accordingly. The optimal vector of parameters $\lambda = [\lambda_1, \lambda_2]$ is estimated using Maximum Likelihood Estimation (MLE), where the likelihood incorporates the log-determinant of the covariance matrix of the normalized data and the Jacobian term of the transformation:

$$L(\lambda) = -\frac{n}{2} log * det\big(Cov(Z)\big) - \sum_{j=1}^{p}(\lambda_j - 1)\sum_{i=1}^{n}\log\,(X_{ij}), \tag{2}$$

where Z is the matrix of normalized values, $Cov(Z)$ is the covariance matrix of Z and the second term is the Jacobian adjustment that accounts for the transformation of variables.

Normal distribution of bivariate data is checked with the Mardia test [24]. The test is based on the measurement of bivariate skewness $\beta_{1,k}$ and kurtosis $\beta_{2,k}$ of the sample:

$$\beta_{1,k} = \frac{1}{N^2}\sum_{i=1}^{N}\sum_{j=1}^{N}[(X_i - \bar{X})^T S_N^{-1}(X_j - \bar{X})]^3, \tag{3}$$

$$\beta_{2,k} = \frac{1}{N}\sum_{i=1}^{N}[(X_i - \bar{X})^T S_N^{-1}(X_i - \bar{X})]^2, \tag{4}$$

where X is a k-dimensional vector of variables, $X = (X_1, X_2, \dots, X_k)$ and S_N is a biased sample variance matrix of the multivariate variable X. It is calculated by a formula:

$$S_N = \frac{1}{N}\sum_{i=1}^{NN}(X_i - \bar{X})(X_i - \bar{X})^T, \tag{5}$$

where $\bar{X}$ is a means vector of the independent variable of the sample, $X = (X_1, X_2, \dots, X_k)^T$.

To assess the multivariate normality of a dataset, the Mardia test employs two separate criteria based on skewness and kurtosis. These conditions must be satisfied for the assumption of multivariate normality to hold.

Test statistic for $\beta_{1,k}$ goes by

$$\frac{n*\beta_{1,k}}{6} \leq \chi^2, \tag{6}$$

where χ^2 denotes the upper quantile of the chi-squared distribution with degrees of freedom $k(k+1)(k+2)/6$ and $\alpha = 0.005$ is a proposed significance level.

For the kurtosis part, the test statistic $\beta_{2,k}$ is compared against the $1-\alpha$ quantile of the normal distribution N with the mathematical expectation $\mu = k(k+2)$ and variance $\sigma^2 = 8k(k+2)/N$

$$\beta_{2,k} \leq N_{1-\alpha}(\mu, \sigma^2). \tag{7}$$

In addition to distributional assumptions, sample quality is improved through the removal of anomalous data points (outliers). The identification of such points is performed using the squared Mahalanobis distance

$$d_i^2 = (Z_i - \bar{Z})S_Z^{-1}(Z_i - \bar{Z}), \tag{8}$$

where $\bar{Z}$ is a means vector of the normalized independent variable of the sample, $Z = (Z_1, Z_2, \dots, Z_k)^T$, S_Z is a biased sample variance matrix for normalized data.

Statistical threshold based on the *F*-distribution, for a confidence level 1 – α, the cutoff value *T* is calculated by

$$T = \frac{k(n^2-1)}{n(n-k)}F_{1-\alpha}(k, n-k), \tag{9}$$

where k is the number of dimensions, n is the number of samples, and $F_{1-\alpha}$ is the quantile function of the *F*-distribution.

If the value of d_i^2, $i = 1,2,\dots,n$ exceeds a statistical threshold based on the *F*-distribution, it is considered an outlier.

After removing all outliers, the ellipse model can be obtained by calculating the mean and covariance matrix from the cleaned data. Based on it, a normalized prediction ellipse can be built, which finds outliers by using the same approach as was described before for formulas (8) and (9).

According to [23], the transformed prediction ellipsoid can be obtained based on the constructed prediction ellipsoid for normalized data using the inverse transformation. In the particular case of two variables, have transformed prediction ellipse is given by the formula

$$\frac{[\psi_1(X_1)-m_{Z_1}]^2}{S_{Z_1}^2} + \frac{[\psi_2(X_2)-m_{Z_2}]^2}{S_{Z_2}^2} - \frac{2S_{Z_1Z_2}[\psi_1(X_1)-m_{Z_1}][\psi_2(X_2)-m_{Z_2}]}{S_{Z_1}^2 S_{Z_2}^2} = \frac{2(N^2-1)(S_{Z_1}^2 S_{Z_2}^2 - S_{Z_1Z_2}^2)}{N(N-2)S_{Z_1}^2 S_{Z_2}^2} F_{2,N-2,a}, \tag{10}$$

where ψ means normalization transformation by bivariate Box-Cox; m_Z is the mean vector $m_Z = (m_{Z_1}, m_{Z_2})^T$; S is the covariance matrix.

The authors have collected the sample dataset of the code metrics of 140 Game Engine Java applications hosted on the GitHub platform [25], which were collected using static source code analysis [26]. 88 metrics were collected from article [27], and another 74 metrics were retrieved from [28].

Overall, the dataset consisted of 302 code metrics. The dataset includes CBO/NCL and RFC/NCL. These metrics can be obtained at an early stage of project planning from the conceptual model of the application.

The descriptive statistic of the initial data set is presented in Table 1.

Table 1
App descriptive statistic (N=302)

Metric Name	Min	Max	Mean	RMSD
NCL	4	7874	540.348	1020.819
CBO/NCL	0.200	22.417	5.806	3.690
RFC/NCL	1.325	37.278	12.648	5.728

6. Experiment

6.1. Normalization, outlier removal, and model building

In this paper, we have extended the results to a larger amount of data of metrics RFC and CBO. As in [23], to detect outliers, we apply the technique based on the squared Mahalanobis distance for the two-dimensional normalized data. To build the transformed prediction ellipse for detecting outliers, we use bivariate normalizing Box-Cox transformations.

The bivariate dataset consisting of software design metrics CBO/NCL and RFC/NCL was initially evaluated for multivariate normality using the Mardia test, which assesses both skewness and kurtosis. The results indicated that the dataset was not normally distributed, which violates a core assumption for statistical modeling and Mahalanobis distance-based prediction ellipse construction.

The Mardia skewness and kurtosis values were calculated using (3) and (4), which are equal to 3.58 and 13.86, respectively.

Test statistic for $\beta_{1,k}$ and $\beta_{2,k}$ exceeded the critical thresholds, confirming significant deviation from multivariate normality based on (6) and (7) comparison, resulting in $180.61 > 14.86$, $12.74 > 2.57$.

To address the non-normality, the dataset was subjected to a multivariate Box-Cox transformation, which requires all values to be strictly positive. Optimal λ parameters for each variable were estimated via maximum likelihood λ = [0.276, 0.227].

During the transformation process, iterative outlier detection and removal were employed using the Mahalanobis distance and a statistical test threshold based on the *F*-distribution using the described logic in (8) and (9):

$$D^2 \leq \frac{k(N^2-1)}{N(N-k)} F_{k,N-k,a} \approx 10.86.$$

Four outliers were removed iteratively, each time re-estimating the Box-Cox parameters and recalculating the Mahalanobis distances. The descriptive statistic of the cleaned data set is presented in Table 2.

Table 2
App descriptive statistic (N=298)

Metric Name	Min	Max	Mean	RMSD
NCL	4	7874	547.161	1025.952
CBO/NCL	0.696	22.417	5.865	3.678
RFC/NCL	3	37.278	12.636	5.609

Emissions in the assumption of normal distribution were also detected. To detect outliers, we used the technique based on the squared Mahalanobis distance without data normalization. During this process, 25 outliers were detected iteratively. This number is more than six times greater than using the data normalization. At the same time, only two data points were detected as the same outliers by both methods.

After clearing all outliers, optimal λ values were found as λ = [0.235, 0.109]. The Mardia skewness and kurtosis values were calculated on cleaned data using (3) and (4), which are equal: 0.135 and 8.162. Test statistic for $\beta_{1,k}$ and $\beta_{2,k}$ are within critical thresholds, indicating no significant deviation from multivariate normality based on the comparison using (6) and (7), resulting in $6.75 \leq 14.86$, $0.351 \leq 2.57$.

The prediction ellipse for normalized data is defined with the next mean vector and covariance matrix:

$$\bar{Z} = [1.98, \quad 2.82];$$
$$S_Z = \begin{bmatrix} 0.82 & 0.32 \\ 0.32 & 0.31 \end{bmatrix},$$

and has the following form:

$$\frac{(Z_1 - 1.98)^2}{0.82} + \frac{(Z_2 - 2.82)^2}{0.31} - \frac{0.64(Z_1 - 1.98)(Z_2 - 2.82)}{0.32} = 6.54$$

These values were used to build a prediction ellipse equation for normalized data from RFC/NCL and CBO/NCL metrics. The constructed prediction ellipse is presented in Figure 1.

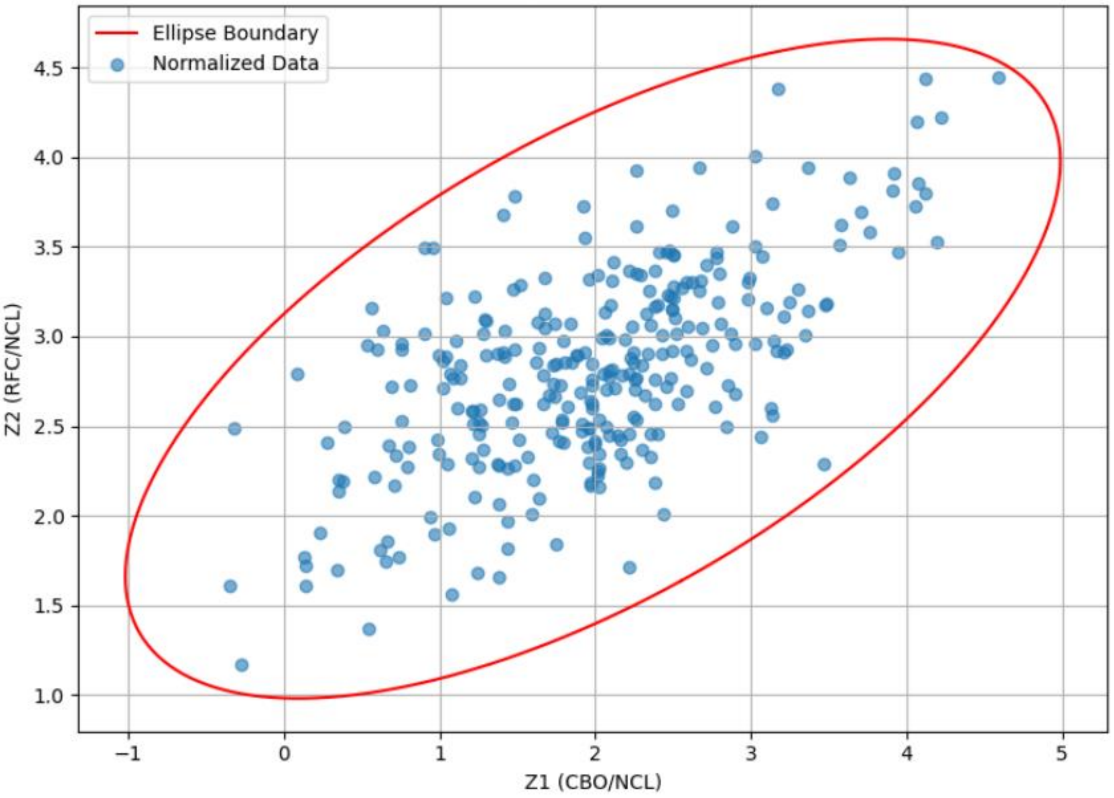

Figure 1: Prediction ellipse for normalized RFC/NCL and CBO/NCL metrics

Based on the approach described by formula (10), it is possible to build the transformed prediction ellipse for initial data:

$$\frac{(\psi_1(X_1) - 1.98)^2}{0.82} + \frac{(\psi_2(X_2) - 2.82)^2}{0.31} - \frac{0.64(\psi_1(X_1) - 1.98)(\psi_2(X_2) - 2.82)}{0.32} = 6.54$$

Described transformed prediction ellipse can be used for analyzing new projects in case of outliers and falling out of range from the given regions of the ellipse.

The transformed prediction ellipse is presented in Figure 2.

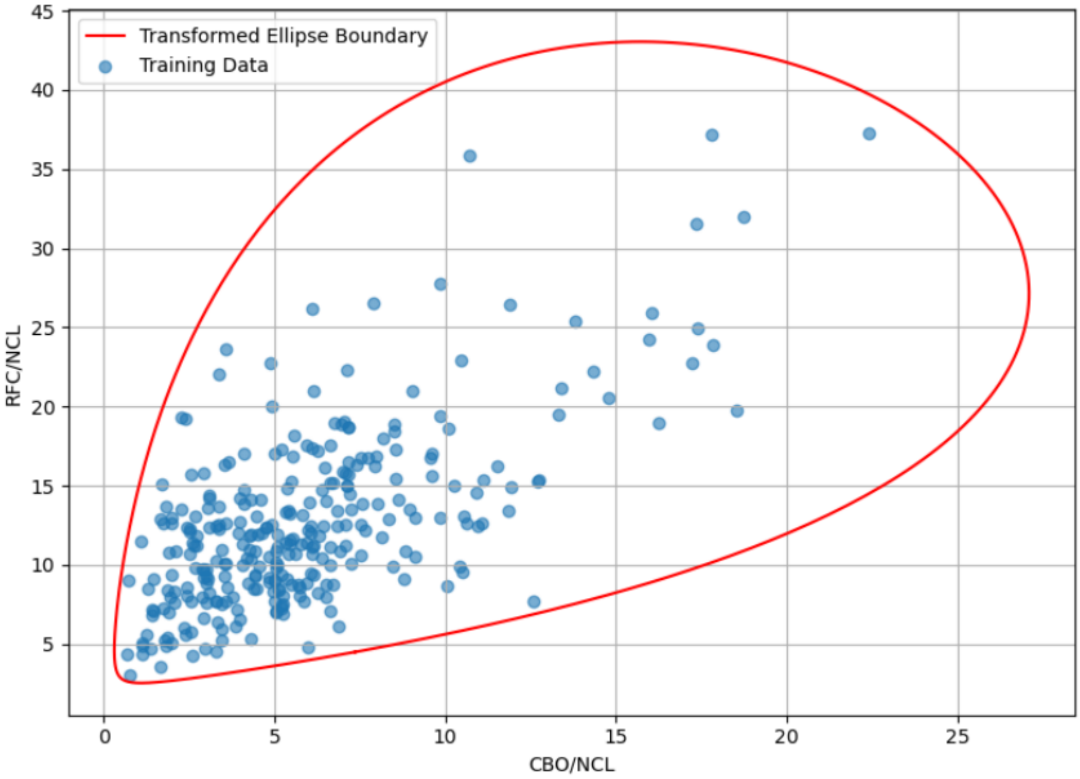

Figure 2: Transformed prediction ellipse for RFC/NCL and CBO/NCL metrics

6.2. Test points

A sample dataset of the code metrics from 26 different Java applications was collected, which were hosted on the GitHub platform [25] and did not participate in the construction of this model. The goal is to determine whether each project falls within the expected boundaries of design complexity defined by a previously built prediction ellipse or if it should be considered an outlier.

These applications vary in size and complexity, ranging from lightweight utility tools and educational examples to full-featured frameworks and production-grade systems. Projects were chosen to reflect a diversity of architectural styles, developer practices, and application scopes within the Java ecosystem.

To test the model, it is necessary to use data within the same range as the data used to build the model. In the original dataset, there was a project with a CBO/NCL value of 0.2, but it was removed as an outlier. In the cleaned data and constructed model, the minimum CBO/NCL value is 0.696, so the test data with CBO/NCL values of 0.25 and 0.5 cannot be estimated using the constructed model. To account for such low CBO/NCL values, a separate model needs to be built. Test data were shown in Table 3 and Figures 3, 4.

Table 3
Test code metrics data of Data Science and Machine Learning Java applications

PROJECT GITHUB URL	NCL	CBO/NCL	RFC/NCL
Anshu231/bookstore.git	26	3.538	8.077
emrekcse/calculator-with-android-studio-Java.git	**4**	**0.250**	**2.250**
google/cel-java.git	774	6.391	16.168
maringallien/Chat-App.git	52	1.615	6.365
mc4/chess-ai.git	27	6.222	15.259
wasabeef/glide-transformations	25	1.000	7.000
glygener/glygen.cfde.generator.git	80	2.613	11.475

snowy-autumn/java-tor.git	31	2.419	8.065
SuperRonJon/JavaAsciiGenerator.git	**2**	**0.500**	**7.500**
SuperRonJon/JavaSudokuSolver	**3**	**0.667**	**10.667**
kroxylicious/kroxylicious.git	1006	3.243	9.877
orhanobut/logger	14	3.429	10.643
elastic/logstash	737	3.927	10.028
projectlombok/lombok	3604	2.341	4.298
PulseBeat02/mcav.git	305	3.334	7.820
dedetive/misle-java.git	94	6.500	17.255
mockito/mockito	1825	3.281	6.642
plantuml/plantuml	3475	8.913	16.137
prestodb/presto.git	1186	4.369	13.683
dernasherbrezon/r2cloud.git	453	5.119	16.450
cadupereira21/recipe-app-java	14	2.143	3.643
QuarkOS/Synapse.git	16	1.438	4.688
BahaaMohamed98/TaskTracker.git	16	2.438	7.375
AR10Dev/TextAnalyzr.git	**2**	**0.500**	**9.500**
code4craft/webmagic	309	3.657	8.460
cptntotoro/yp-online-store-showcase.git	79	3.418	4.544

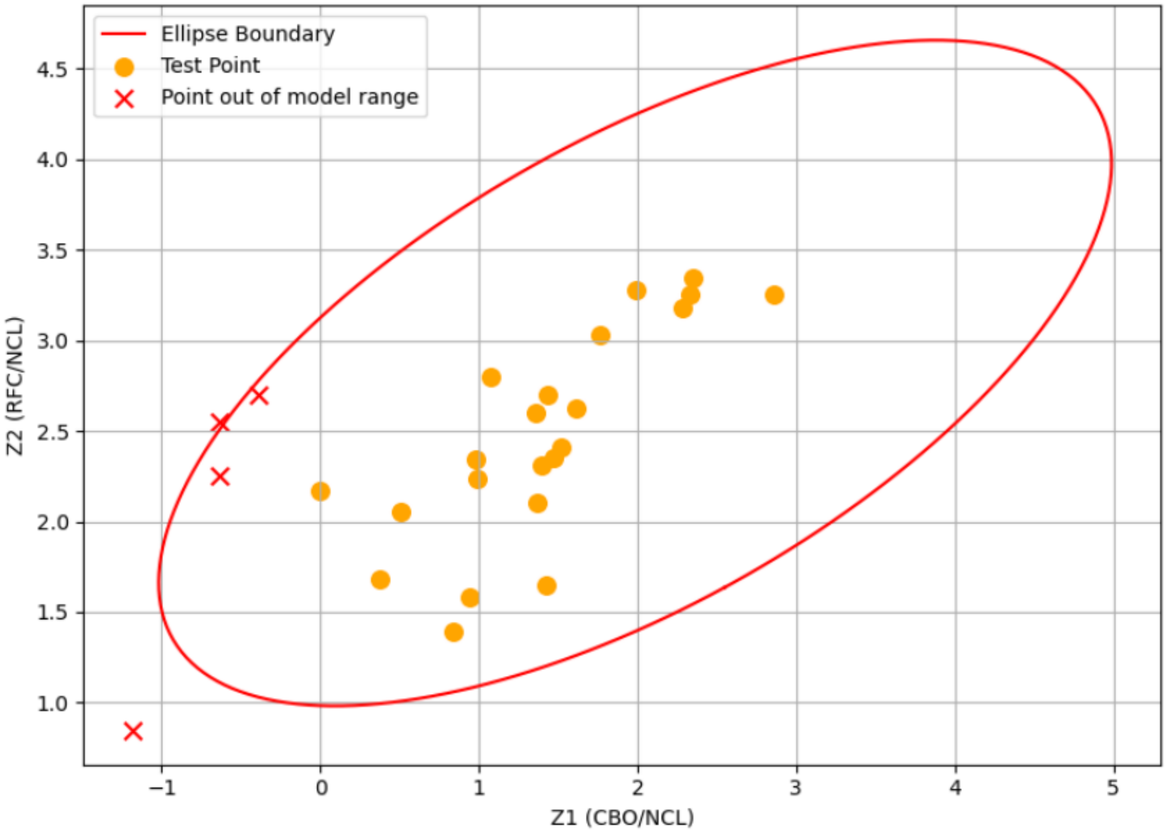

Figure 3: Prediction ellipse for normalized RFC/NCL and CBO/NCL test metrics

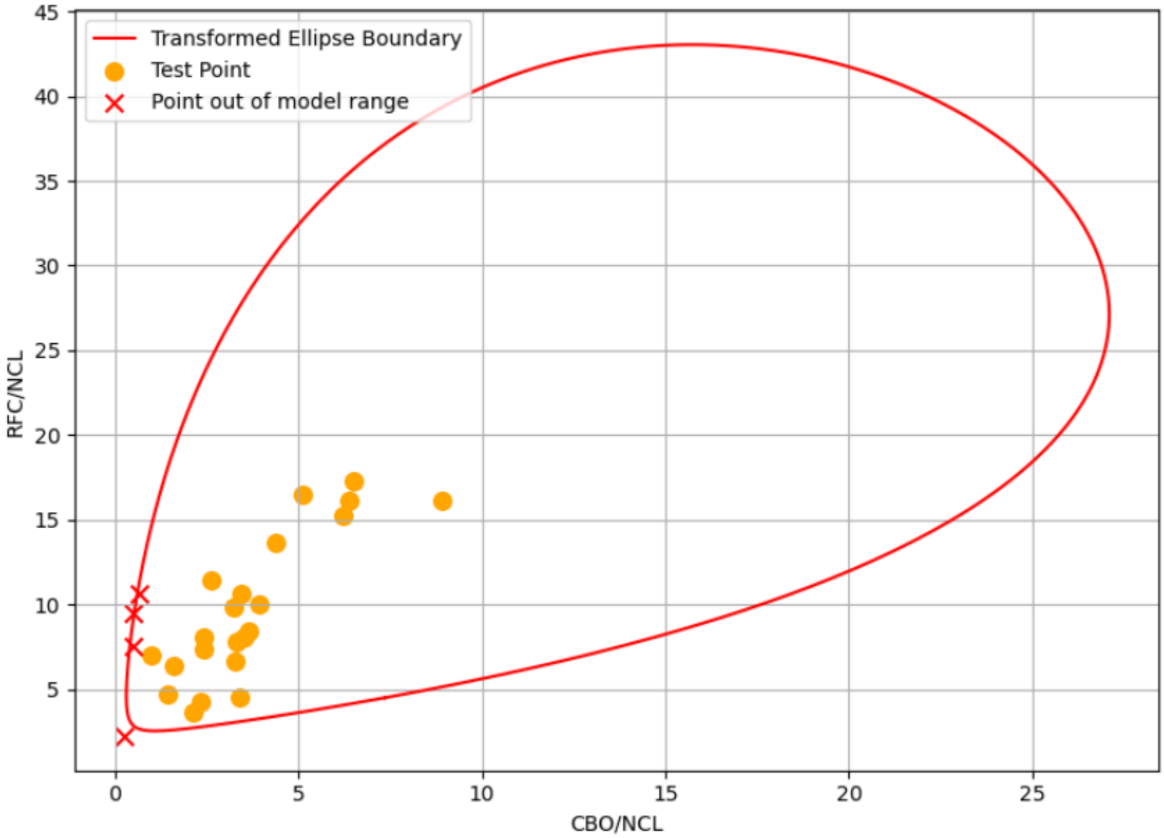

Figure 4: Transformed prediction ellipse for RFC/NCL and CBO/NCL test metrics

All of the analyzed projects within the boundaries of the transformed prediction ellipse incline towards the left side of the ellipse without any entries in the right one. However, the test data, which was used, has a range of NCL [4, 7874], RFC [3.642, 17.255], and CBO [1, 8.912] that correlates with training data, but doesn't fully cover its whole range.

7. Conclusion

To detect the outliers in the two-dimensional software metrics RFC and CBO, have proposed to apply mathematical model in the form of a transformed prediction ellipse based on their normalization using the bivariate Box-Cox transformation. The number of classes is not used directly, but indirectly through the RFC divided by NCL and CBO divided by NCL. The practical application of this model is detecting outliers in the two-dimensional data of software metrics RFC and CBO from applications in Java.

Advantages of this model are that it can be used for object-oriented metrics, such as RFC and CBO, that do not follow a normal distribution.

Disadvantages of this model are that it can be applied only to Java applications.

Limitations of this model are that it doesn't cover the whole range of possible values of the object-oriented metrics RFC and CBO, and can be used for the next range of metric values: NCL [4, 7874], RFC [3, 37.278], CBO [0.696, 22.417]. These limitations are due to the range of metric values that were used to build the model.

Moving forward, we plan to develop a mathematical model in the form of a transformed prediction ellipse for detecting outliers in the two-dimensional data of the software metrics RFC and CBO that do not have limitations due to the programming language and the sample size.

Declaration on Generative AI

During the preparation of this work, the authors used OpenAI ChatGPT-4 in order to: Text translation; Grammar and spelling check. After using this tool/service, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

[1] D. Mendez, P. Avgeriou, M. Kalinowski, N. bin Ali (Ed.), Handbook on teaching empirical software engineering. Cham: Springer, 2024.

[2] P. Anil, G. Manjari, Software Metrics Selection for Fault Prediction: A Review, International Journal of Management, Technology and Engineering 8 (2018) 1267–1283.

[3] M.M. NezhadShokouhi, M.A. Majidi, A. Rasoolzadegan, Software defect prediction using over-sampling and feature extraction based on Mahalanobis distance, The Journal of Supercomputing 76 (2020) 602–635. doi: 10.1007/s11227-019-03051-w.

[4] YS. Seo, DH. Bae, On the value of outlier elimination on software effort estimation research, Empirical Software Engineering 18 (2013) 659–698. https://doi.org/10.1007/s10664-012-9207-y.

[5] S.R. Chidamber, C.F. Kemerer, Towards a metrics suite for object oriented design, ACM SIGPLAN Notices 26 11 (1991) 197–211. doi: 10.1145/118014.117970.

[6] Y. Suresh, J. Pati, S. Ku Rath, Effectiveness of software metrics for object-oriented system, Procedia Technology 6 (2012) 420–427. doi: 10.1016/j.protcy.2012.10.050.

[7] R. Shatnawi, Empirical study of fault prediction for open-source systems using the Chidamber and Kemerer metrics, IET Software 8 3 (2014) 113–119. doi: 10.1049/iet-sen.2013.0008.

[8] A.-J. Molnar, A. Neamţu, S. Motogna, Evaluation of Software Product Quality Metrics, CCIS, 1172 (2020) 163–187. doi: 10.1007/978-3-030-40223-5_8.

[9] I.M.A. Wikantyasa, A.P. Kurniawan, S. Rochimah, CK Metric and Architecture Smells Relations: Towards Software Quality Assurance, in: Proceedings of the 14th International Conference on Information and Communication Technology and System (ICTS), 2023, pp. 13–17. doi: 10.1109/ICTS58770.2023.10330874.

[10] S. Jin, Z. Li, B. Chen, B. Zhu, Y. Xia, Software Code Quality Measurement: Implications from Metric Distributions, in: Proceedings of the IEEE International Conference on Software Quality, Reliability and Security, QRS, 2023, pp. 488–496. doi: 10.1109/QRS60937.2023.00054.

[11] J. Härtel, R. Lämmel, Operationalizing validity of empirical software engineering studies, Empirical Software Engineering 28 6 (2023) 153. doi: 10.1007/s10664-023-10370-3.

[12] R.A. Johnson, D.W. Wichern, Applied Multivariate Statistical Analysis, New Jersey: Pearson Prentice Hall, 2007.

[13] J.W. Osborne, Best Practices in Data Cleaning: A Complete Guide to Everything You Need to Do Before and After Collecting Your Data, SAGE Publications, Inc., 2013.

[14] T. Etherington, Mahalanobis distances for ecological niche modelling and outlier detection: implications of sample size, error, and bias for selecting and parameterising a multivariate location and scatter method, PeerJ, 9, 2021, pp. e11436. doi: 10.7717/peerj.11436.

[15] S. Prykhodko, N. Prykhodko, L. Makarova, K. Pugachenko, Detecting outliers in multivariate non-Gaussian data on the basis of normalizing transformations, in: Proceedings of the IEEE First Ukraine Conference on Electrical and Computer Engineering (UKRCON), 2017, pp. 846–849. doi: 10.1109/UKRCON.2017.8100366.

[16] M.G. Al-Obeidallah, The Impact of Design Patterns on Software Maintainability and Understandability: A Metrics-based Approach, ICIC Express Letters, Part B: Applications, Vol. 12, No.12, 2021, pp. 1111–1119. doi: 10.24507/icicelb.12.12.1111.

[17] Monika, P. Sharma, Prediction of Fault-Proneness using CK Metrics, International Journal of Computer Science and Information Technology Research, Vol. 4, Issue 3, 2016, pp. 114–118.

[18] T. Filó, M. Bigonha, K. Ferreira, A Catalog of Thresholds for Object-Oriented Software Metrics, in: Proceedings of the First International Conference on Advances and Trends in Software Engineering, 2015, pp. 48–55.

[19] I. Turnu, G. Concas, M. Marchesi, R. Tonelli, Entropy of some CK metrics to assess object-oriented software quality, International Journal of Software Engineering and Knowledge Engineering 23 2 (2013) 173–188. doi: 10.1142/S0218194013500034.

[20] M. Friendly, G. Monette, J. Fox, Elliptical Insights: Understanding Statistical Methods Through Elliptical Geometry, Statistical Science, 28 (2013) 1–39. doi:10.1214/12-STS402.

[21] F. Golestaneh, P. Pinson, R. Azizipanah-Abarghooee, H.B. Gooi, Ellipsoidal Prediction Regions for Multivariate Uncertainty Characterization, IEEE Transactions on Power Systems, 33 4, 2018, pp. 4519–4530. doi: 10.1109/TPWRS.2018.2791975.

[22] S. Prykhodko, N. Prykhodko, L. Makarova, A. Pukhalevych, Outlier Detection in Non-Linear Regression Analysis Based on the Normalizing Transformations, in: IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), 2020, pp. 407–410. doi: 10.1109/TCSET49122.2020.235464.

[23] S. Prykhodko, L. Makarova, K. Prykhodko, A. Pukhalevych, Application of Transformed Prediction Ellipsoids for Outlier Detection in Multivariate Non-Gaussian Data, in: Proceedings of the IEEE 15th International Conference on Advanced Trends in Radioelectronics, Telecommunications and Computer Engineering (TCSET), 2020, pp. 359–362, doi: 10.1109/TCSET49122.2020.235454.

[24] K.V. Mardia, Measures of multivariate skewness and kurtosis with applications, Biometrika, volume 57, 1970, pp. 519–530. doi: 10.1093/biomet/57.3.519.

[25] Build and ship software on a single, collaborative platform, 2025. URL: https://github.com.

[26] SourceMeter for Java, 2025. URL: https://sourcemeter.com.

[27] S. Prykhodko, L. Makarova, A. Pukhalevych, Statistical Analysis of the Three-Dimensional Data of Software Metrics RFC, CBO, and WMC that are not Normally Distributed, in: Proceedings of International Conference on Applied Innovation in IT, vol. 13, issue 1, 2025, pp. 127–132. doi: 10.25673/119254.

[28] O. Oriekhov, T. Farionova, L.S. Chernova, L. Chernova, M. Vorona, Nonlinear regression models for software size estimation of Data Science and Machine Learning Java-applications, in: Proceedings of the 5th International Workshop IT Project Management (ITPM), 2024; CEUR Workshop Proceedings, Vol. 3709, 2024, pp. 54–66. doi:10.23939/IW_itpm2024.054.