scientific reports

Check for updates

OPEN

An empirical analysis on webservice antipattern prediction in different variants of machine learning perspective

Lov Kumar[1]✉, Sahiti Tummalapalli[2], Lalita Bhanu Murthy[2], Sanjay Misra[3]✉ & Aneesh Krishna[4]

Anti-patterns are explicit structures in the design that represents a significant violation of software design principles and negatively impacts the software design quality. The presence of these Anti-patterns highly influences the maintainability and perception of software systems. Thus it becomes necessary to predict anti-patterns at the early stage and refactor them to improve the software quality in terms of execution cost, maintenance cost, and memory consumption. In the anti-pattern prediction domain, during research analysis, it was realized that there had been very little work instigated on addressing both class imbalance and feature redundancy problems jointly to enhance models' performance and prediction accuracy. It has been perceived in the literature survey to study droughts with a comprehensive comparative analysis of different sampling and feature selection strategies. To achieve greater precision results and performance, this research constructs a web service anti-pattern prediction model over preprocessed software source code metrics using sampling and feature selection techniques to handle imbalanced data and feature redundancy to gain flawless web service anti-pattern prediction outcomes. Considering the above erudition, we have applied different variants of aggregation measures to find the metrics at the system level. These extracted metrics are used as input, so we have also applied different variants of feature selection techniques to remove irrelevant features and select the best combination of features. After finding important features, we have also applied different variants of data sampling techniques to overcome the problem of class imbalance. Finally, we have used thirty-three different classifiers to find import patterns that help identify anti-patterns. These all techniques are compared using Accuracy and Area Under the ROC (receiver operating characteristic curve) Curve (AUC). The experimental result of web service anti-pattern prediction models validated on 226 WSDL files illustrates that the least square support vector machine (LSSVM) with RBF kernel attains the best performance among the other 33 competing classifiers employed with the lowest Friedman mean rank value of 1.18. During comparative analysis over different feature subset selection techniques, the outcome indicates the mean accuracy value of 88.40% and mean AUC value of 0.88 for the models developed using significant features are higher in comparison to other techniques. The result shows the up-sampling methods (UPSAM) method secured the highest mean accuracy % and mean AUC with values of 86.14% and 0.87, respectively. The experimental result indicates the performance of the web service anti-pattern prediction models is adversely impacted by class imbalance and irrelevance of features. The outcome demonstrates that the performance of trained models improved with an AUC value between 0.805 to 0.99 post-application of sampling and feature selection strategies without using feature selection and sampling techniques. The outcome implies that USMAP achieves better performance. The result demonstrates that the models developed using significant features drive the desired effect compared to other implemented feature selection techniques.

Keywords Data sampling, Feature selection techniques, Anti-pattern, Aggregation measure, Machine learning

[1]NIT Kurukshetra, Kurukshetra, Haryana, India. [2]BITS Pilani, Hyderabad, India. [3]Institute for Energy Technology (IFE), Halden, Norway. [4]Curtin University, Perth, Australia. ✉email: lovkumar505@gmail.com; sanjay.misra@ife.no

System autonomy, heterogeneity, and context adaptability are critical in the software business, leading to the development of web services based on service-oriented architecture (SOA). For successful businesses and contemporary governments, SOA is the progression of distributed computing toward integrating expert departments and IT. Services may be accessed via the internet using the web service implementation of SOA, which is agnostic of the platform and programming language. SOA is generally regarded in IT systems as the technology that can improve the receptivity of both business and IT organizations since it is self-adaptable to context. Web services may be built in various languages and on various platforms, allowing them to be used on a wide range of devices.

Modeling Service-Based Systems (SBSs) like Paytm, DropBox and Amazon are made feasible by SOA, and the growth of these systems causes many challenges. As new devices and technologies are introduced, SBSs must evolve to keep up with the demands of their users. Like any other big and complicated system, SBSs are prone to ongoing modification to accommodate new user needs and modify the execution circumstances. It's also possible that all of these modifications may decrease SBS' Quality of Service (QoS) and result in a retro design, which has been given the name of "Anti-patterns"[1]. Structures like these imply a breach of fundamental design principles and a decrease in design quality. Because they make it challenging to improve and maintain a software system, anti-patterns are helpful for spotting issues with its design, source code, or overall project management. Therefore, it has become compulsory to develop prediction models that help to detect anti-patterns present in web services. Software quality researchers have used simple models to predict different types of anti-patterns based on source code metrics that help improve the software quality in terms of execution cost, maintenance cost, and memory consumption. Empirical experiments have been carried out in the past related to web service anti-pattern predictions (Travassos et al.[2], Marinescu et al.[3], Munro et al.[4], Ciupke et al.[5] Simon et al.[6], Rao et al.[7], Khomh et al.[8], Moha et al.[9]). Though these research works have raised the need to develop perdition models, it was realized that there had been very little work instigated on addressing both class imbalance and feature redundancy problems jointly to enhance models' performance and prediction accuracy. It has been perceived in the above work to study droughts with a comprehensive comparative analysis of different sampling and feature selection strategies.

In this work, we investigate the predictive power of different aggregation measures which are used for finding file-level metrics, feature selection techniques that are used for selecting significant features, data sampling techniques that are used for handling the class imbalance nature of datasets, and different variants of machine learning for finding the pattern. Here, our focus is on how accurately these techniques help to predict anti-patterns present in web services. Initially, we selected 226 different web-service as WSDL from various domains such as finance, tourism, health, education, etc. Then we applied the WSDL2Java tool to each WSDL file to extract the java files. After extracting the java files, we have used CKJM[10] tool proposed by Chidamber and Kemerer to find metrics at the class level. Since our objective is to find the anti-pattern present in the WSDL file, so we have applied different variants of aggregation measures to find metrics at the system level. After computing metrics at the system level, we have also applied feature selection techniques to find the significant set of features, which are later used as input for the anti-pattern prediction models. We also observed that the considered data have imbalanced nature of classes. Henceforth, to handle the class imbalance problem and its impact on the prediction accuracy of the models, we have also used five data sampling techniques. We compare the performance of the models generated using this sampling technique with the model developed using the original data (ORGD).

Finally, we have applied different categories of machine learning techniques to find import patterns that help to identify anti-patterns present in unseen WSDL files. Initially, we have applied the most frequently used classifiers like different variants of Naive Bayes (Bernoulli, Gaussian, Multinomial), decision trees, logistic regression, support vector machines with different kernels, and artificial neural networks with different back-propagation algorithms. Different researcher mainly uses these types of classifiers to predict software quality parameters. Then, advanced levels of classifiers like least square support vector machines with multiple kernels and extreme and weighted extreme learning machines with multiple kernels have been used to find better sets of patterns for anti-pattern predictions. Finally, we have used ensemble learning and deep-learning approaches to find the best patterns for anti-pattern predictions. The predictive power of these techniques is evaluated in terms of accuracy and AUC values and validated with 5-fold cross-validation approaches on 226 different web-service. In order to find the significant impact of the techniques, we have used Wilcoxon Signed Rank Test (WSRT) with Friedman mean rank (FMR).

The major contributions of this research work are:

- Proposed a framework to predict web service anti-patterns based on extracted java files of WSDL.
- Proposed a framework using the aggregation measures concept to extract file-level metrics from class-level metrics.
- Usage of different sampling approaches to counter the class imbalance problem.
- Usage of different feature selection techniques to remove irrelevant features and set the right sets of features.
- thirty-three different classifiers are considered to develop a model to identify the files with anti-patterns.
- Various statistical tests were conducted to determine the effectiveness of the proposed anti-pattern detection model.The paper is organized as follows: Section 2 provides the summary of related work in the field of software fault prediction. Section 3 explains the used methodologies in our experimentation. The research framework, result analysis, and model performance is presented in Sections 4 and 5. Section 6 covers the comparative analysis. The final results discussion and conclusion work are depicted in Sections 7 and 8 respectively.

Related work

There is a good number of existing methods proposed by various researchers to predict anti-patterns or code smells present in object-oriented software. A manual procedure to identify anti-pattern or design smells is proposed by Travassos et al.[2]. They have used manual reviews and reading techniques types of concepts to find the smells that do not meet the specification. A similar kind of work is also proposed by Marinescu et al.[3] to predict the design smell present in software systems based on extracted metrics from the source code of the software system. They have executed their proposed work on the IPLASMA tool with the help of some detection techniques to find the pattern that helps to identify smells in a software system. They have applied ten detection techniques to predict anti-patterns or code smells. The major limitations of their approach are that extensive knowledge of metric-based rules is required to detect an anti-pattern successfully, and the varied threshold values lead to a varied outcome. Munro and his team[4] also proposed one new method with the objective to overcome the limitations of text-based descriptions for predicting systematically characterized code smells. They have applied metric-based heuristics concepts to detect anti-patterns.

Ciupke et al.[5] presented a method to study legacy code by specifying design problems as queries. Their approach is based on extracting the occurrences of the problems using models designed using extracted metrics from the source code of software systems. Simon and his team[6] proposed methods based on visualization concepts to find the correlation between fully automated approaches, which are productive, systematic, and time-consuming. The major advantage of their strategies is there is no need for effective manual inspections.

Rao et al.[7] introduced a method to propose anti-patterns based on the Design Propagation Probability concept to design the models that will treat like detection techniques. Based on the design Propagation Probability concept, they have focused on two anti-patterns, such as Divergent change and Shotgun surgery. Similarly, Khomh and his team[8] presented the method with the help of anti-pattern definition, Goal Question Metric(GQM), and Bayesian Detection Expert(BDTEX) to develop Bayesian Belief Networks(BBN). The BBN method allows quality analysts to use their prior probability to predict anti-patterns.

Moha et al.[9] proposed an automated method to predict different types of anti-patterns like Spaghetti Code, Functional Decomposition, Blob, and Swiss Army Knife. Their proposed methods also help to identify 15 underlying code smells. They gave the DECOR name of their proposed methods containing all the necessary steps used to specify and detect code and design smells. Their team also proposed another detection method called DETEX[9] which helped to provide a platform to convert the rules extracted from the DECOR method into detection algorithms. They have clearly explained the correlation between the metrics extracted from code with different categories of anti-patterns.

Hemanta Kumar Bhuyan and Vinayakumar Ravi presented the importance of feature selection techniques in data mining applications[11]. They have proposed the optimization model using a Lagrangian multiplier to find and analyze a new class. They have used several classifiers with searching and statistical methods to validate the proposed subfeatures. Their finding confirms that their proposed methods benefit novel classes based on selected subfeature data. Hemanta Kumar Bhuyan and Narendra Kumar Kamila also provide the content related to the importnace of the feature selection techniques in data mining applications[12]. The have used fuzzy probabilities to proposed privacy preservation of individual data for both feature and sub-feature selection. They conclused that the fuzzy random variable approach confined the expected range on which the selection of sub-feature from feature database is made easy. Similar work is also done by Hemanta Kumar Bhuyan et al. to find the importance of feature selection during model development[13]. They proposed methods to choose the optimal feature for classification by utilizing mutual information (MI) and linear correlation coefficients (LCC). Their proposed methods offers the best selection on the same data set as compared to others.

Motivation

Based on the above survey, profound research has been conducted in the area of web service anti-pattern prediction models using machine learning approaches. However, further analysis indicates there is very little investment seen in converting file-level metrics using class-level, handling class imbalance of datasets, removing irrelevant features, and comparing wide varieties of machine learning techniques. As a result, there is a need for in-depth research to evaluate the performance of anti-pattern prediction models by combining aggregation, feature selection, and sampling techniques. This point is our primary motivation for our present work. It leads us to endow our focus on implementing the proposed model to address the substantial gap identified to extemporize the performance and predictability of the anti-pattern prediction model by engaging aggregation, sampling, and feature selection techniques jointly with a wide variety of machine learning techniques. This research work exploits the implication of sixteen aggregation measures, seven feature selection techniques, five sampling strategies, and thirty-three different classifiers to develop the best web service anti-pattern prediction models. The performance of these developed models is analyzed using AUC and Accuracy metrics. This leads to the contextual following research questions (RQ):

RQ 1: *Can web-service anti-patter prediction models be developed using source code metrics and machine learning?*
RQ 2: *What is the significant impact of considering reduced sets of features as input on the performance of models?*
RQ 3: *What is the significant impact of sampling techniques on the predictability of anti-pattern prediction models?*
RQ 4: *What effect do different classifiers have on predicting anti-patterns using source code metrics?*

Methodologies

This section enlightens on the components required for our study. We are providing information on datasets, feature selection techniques, sampling strategies, and classification approaches.

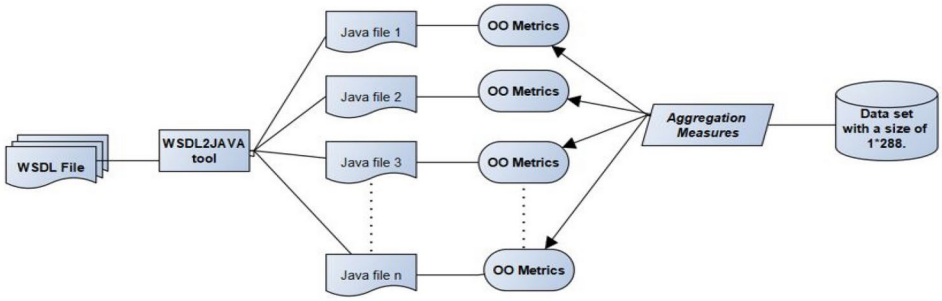


Fig. 1. Pre-processing of dataset.

OO-metrics	Metrics-explanation
Ca: Afferent coupling	The number of classes using the features defined inside a given class
Avg-CC: Average cyclomatic complexity	Mean complexity of methods defined inside class
AMC: Average method complexity	Mean size of methods defined inside class
CAM: Cohesion among methods of class	Ratio of the sum of parameters of methods and product of unique parameters of methods
CBM: Coupling between methods	Total number of methods linked with inherited methods
CBO: Coupling between object classes	The number of classes linked with a given class
DAM: Data access metric	Ratio of protected or private attributes and the total number of attributes
DIT: Depth of inheritance tree	Max depth of tree
Ce: Efferent coupling	The number of classes that a specific class uses
IC: Inheritance coupling	The number of parent classes with which a given class is associated.
LCOM: Lack of cohesion in methods	The number of methods in a class that are unrelated despite the fact that some of the class's fields are shared
LCOM3	Methods lack of cohesion. Henderson-Sellers version
LOC: Lines of code	The number of lines in the source code's text
MAx-CC: Max cyclomatic complexity	Maximum cyclomatic complexity of a class's methods
MOA: Measure of aggregation	Number of data declarations (class fields) with user-defined class types
MFA: Measure of functional abstraction	The ratio of the number of methods inherited by a class to the total number of methods accessible by the class's member methods
NOC: Number of children	Number of immediate descendants of the class
NPM: Number of public methods	Number of methods defined as public inside class
RFC: Response for a Class	Number of unique methods executed after receiving message
WMC: Weighted methods per class	Summation of methods complexity defined inside class

Table 1. Object-oriented software project datasets.

Data collection

We have prepared the datasets in this experiment to validate our proposed anti-pattern prediction model framework. Figure 1 shows the working procedure to prepare datasets. Initially, we applied the WSDL2Java tool on the WSDL file to extract the java files. These extracted java files are used as an input of CKJM[10] tool to find object-oriented metrics as mentioned in Table 1 at the class level. CKJM takes java files as an input and computes metrics at the class level, but we need metrics at the system level because, in the experiment, we predict the anti-pattern at the WSDL level. To achieve this, we have applied aggregation measures to find metrics at the system level. Vasilescu et al.[14] suggested using multiple aggregation measures to find metrics at the higher level without losing information. They have empirically proved that the use of a single aggregation measure creates a data loss problem. So, in this work, we have applied 16 aggregation measures as mentioned in Table 2 on class-level metrics to find metrics at the system level.

Experimental dataset

This experiment makes use of publicly available web-services datasets consisting of 226 WSDL files shared by Ouni et al. on GitHub https://github.com/ouniali/WSantipatterns. Table 3 shows the a detailed description of the considered datasets in terms of different types of anti-patterns. The first column of the table contains the name of anti-patterns like Fine-Grained anti-pattern (FGWS), Chatty anti-pattern (CSW), God Object anti-pattern (GOWS), Data ant-pattern (DWS), Ambiguous Anti-pattern (AWS). The second column contains the number web-service not having these patterns, the third column contains the number web-service having these patterns, and the last column contains the percentage of web-service having these patterns. From Table 3, we can say that the 13 web-service has FGWS anti-pattern with 5.75 %.

Aggregation measure	Computation formula
Variance(ag1)	$var(p) = \frac{\sigma_p}{\mu_p}$
Arithmetic Mean(ag2)	$\mu_p = \frac{1}{R}\sum_{q=1}^{R} p_q$
Skewness(ag3)	$\gamma_1 = \frac{\sum_{q=1}^{R}(p-\overline{p})^3/R}{(\sigma(p))^3}$
Minimum	–
Median(ag4)	$M_p = \begin{cases} p_{R+1/2} & if R is odd \\ 1/2(p_{R/2} + p_{R+2/2}) & otherwise \end{cases}$
Quartile1(25%)(ag5)	–
Theli Index (ag6)	$I_{Theli}(p) = \frac{1}{R}\sum_{q=1}^{R}(\frac{p_q}{\mu_s} * \ln(\frac{p_q}{\mu_s}))$
Standard Deviation (ag7)	$\sigma_p = \sqrt{\frac{1}{R}\sum_{q=1}^{R}(p_q - \mu_q)^2}$
Quartile3(75%) (ag8)	–
Generalized Entropy (ag9)	$GE_p = -\frac{1}{R\alpha(1-\alpha)}\sum_{q=1}^{R}[(\frac{p_q}{\mu_p})^{\alpha} - 1], \alpha = 0.5$
Maximum (ag10)	–
Gini Index(ag11)	$I_{Gini}(p) = \frac{2}{R\sum_p}[\sum_{q=1}^{R}(p_q * q) - (R+1)\sum_p]$
kurtosis (ag12)	$\gamma_2 = \frac{\sum_{q=1}^{R}(p-\overline{p})^4/R}{(\sigma(p))^4}$
Hoover Index	$I_{Hoover}(p) = \frac{1}{2}\sum_{q=1}^{R}\lvert\frac{p_q}{\sum_p} - \frac{1}{R}\rvert$
Atkinson Index(ag13)	$I_{Atkinson}(p) = 1 - \frac{1}{\mu_p}(\frac{1}{R}\sum_{q=1}^{R}\sqrt{p_q})^2$
Shannon Entropy	$E_p = -\frac{1}{R}\sum_{p=1}^{R}[\frac{freq(p_q)}{R} * \ln\frac{freq(p_q)}{R}]$

Table 2. Aggregation measures.

Anti-pattern	NAP	AP	%AP
CWS	205	21	9.29
FGWS	213	13	5.75
AWS	202	24	10.62
GOWS	205	21	9.29
DWS	212	14	6.19

Table 3. Datasets.

Data balancing techniques

The information in Table 3 confirms that the considered datasets have no equal distribution of anti-patterns, i.e., only 9.29% of WSDL files have CSW type of anti-pattern. This information confirms that the considered datasets have a class imbalance problem. So, we have applied five data sampling techniques as Adaptive Synthetic Sampling Technique (ADASYN), Synthetic Minority Oversampling Technique (SMOTE), SVMSMOTE, Borderline SMOTE (BLSMOTE), and UP sampling Technique (UPSAM), to generate balanced data. The predictive ability of these techniques is also compared using the model trained on original data to find the impact of using sampling techniques.

- *SMOTE*[15]: The concept of SMOTE is based on nearest neighbors. It will generate minority class instances.
- *Borderline smote (BLSMOTE)*[16]: BLSMOTE creates new instances of the minority class utilizing the closest neighbors of these cases in the border region between classes.
- *SVM-SMOTE (SVMSMOTE)*[17]: SVMSMOTE generates new minority class samples over the border with SVM to establish a boundary line between the classes using SVM[18].
- *Adaptive synthetic sampling technique (ADASYN)*[19]: ADASYN is built on the notion of adaptively producing minority data samples depending on their distributions. More synthetic data is created for minority-class samples that are more difficult to learn than for minority-class samples that are simpler to understand. This strategy helps to lessen the learning bias imposed by the initial unbalanced data distribution. Still, it may also adaptively move the decision boundary to concentrate on samples that are harder to learn, which is very useful when dealing with large datasets. The most significant distinction between SMOTE and ADASYN is how synthetic sample points for minority data points are generated in each system[20]. In ADASYN, we consider a

density distribution r_x, which determines the number of synthetic samples to create for a given point, while in SMOTE, all minority points have the same weight.
- *Upsampling (UPSAM) technique:* Upsampling is the technique in which the instances from the minority class are randomly duplicated[21].

Selection of relevant metrics

In the process of Knowledge Data Discovery (KDD), Feature Selection (FS) is a vital part of the pre-processing step. Some of the numerous names given to Feature Selection Algorithms include Attribute Selection, Instance Selection, Data Selection, Feature Construction, Variable Selection, and Feature Extraction, to mention just a few. They are primarily used to remove unnecessary and redundant material. Feature selection methods[22] enhance the quality of data and boost data mining algorithms' accuracy by minimizing the data's complexity in terms of space and time. Eliminating duplicate and irrelevant data is the primary goal of feature selection. Several feature selection methods have been released in the last decade; however, the vast majority of them do not perform well on high-dimensional datasets with a significant number of duplicated features. As a result, feature selection is more critical in eliminating irrelevant features[2324]. As a result, machine learning algorithms can concentrate on the features required to build a classification model. Two subclasses of feature selection techniques can be generally distinguished:

- *Metrics selection using feature ranking techniques:* In this technique, each feature is ranked according to a few key criteria before some features that are appropriate for a particular project are chosen.
- *Metrics selection using feature subset selection techniques:* In feature subset selection, our objective is to find a subset of features that have strong predictive power

Metrics selection using feature ranking techniques

- *Selection of significant features(SIGF)* Initially, we applied hypothesis testing to each metric to find "whether the metric can differentiate the WSDL file having anti-pattern or not"[25]. So, In this experiment, we have applied the Wilcoxon signed-rank test at a 0.05 level to find the difference between the metric values for a file having an anti-pattern and not having an anti-pattern. This test is mainly used to find whether two dependent samples are significantly the same or different.
- *Features ranking using information gain (INFG)* An attribute ranking approach that's both simple and quick is widely employed in text classification applications when the sheer volume of data makes it impossible to utilize more complicated methods[26]. If P is an attribute and Q is a class, then eq. 1 and 2 provide the values for the entropy of the class before and after the attribute is observed:

$$H(Q) = -\sum_{q \epsilon Q} p(q) \log_2 p(q) \tag{1}$$

$$H(Q|P) = -\sum_{p \epsilon P} p(p) \sum_{q \epsilon Q} p(q|p) \log_2 p(q|p) \tag{2}$$

When the entropy of a class lowers by a certain level, it indicates how much new information about that class has been supplied by the attribute, which is referred to as information gain.

Based on the information gain value between the class and each P_i, a score is awarded to each P_i:

$$\begin{aligned} IG_i &= H(Q) - H(Q|P_i) = H(P_i) - H(P_i|Q) \\ &= H(P_i) + H(Q) - H(P_i, Q) \end{aligned} \tag{3}$$

- *Features ranking using gain ratio (GNR)* The gain ratio is a modification of the information gain that decreases the bias of the information gain. When picking an attribute, the gain ratio considers the number and size of branches[27]. When the intrinsic information is taken into account, it corrects the information gained. Intrinsic information is the entropy of instance distribution into branches, i.e., how much information is required to determine which branch an instance belongs to. The value of an attribute decreases as the number of intrinsic information increases.

$$\text{GNR} = \frac{\text{Gain of attribute}}{\text{intrisinc information of attribute}} \tag{4}$$

- *Features ranking using OneR attribute evaluation (OneR)* OneR, short for "One Rule," is a straightforward but accurate classification algorithm that generates one rule for each predictor in the data and then selects the rule; with the slightest total error as its "one rule." A rule for a predictor is created by creating a frequency table for each predictor and comparing it to the objective (the target)[28]. Compared to state-of-the-art classification algorithms, it has been shown that OneR creates rules that are only marginally less accurate while also making straightforward rules for people to comprehend.

- *Features extraction using principal component analysis (PCA)* Principle Component Analysis(PCA)[29] is applied to find the new values of features with high variance. The concept is based on removing highly correlated features and finding new sets of feature values. Here, we have applied PCA with the varimax rotation technique on extracted sets of file-level source code metrics. In this work, we have considered all the Principle Components whose eigenvalue is greater than 1.

Metrics selection using feature subset selection techniques

- *Selection of features using correlation coefficient (CORR)* Correlation Coefficient feature selection is used to remove the features having high co-relation with other features. In this paper, we have used the concept of Pearson's correlation to find the pair of features having highly correlated or not, i.e., >=0.7 or <=-0.7 represent the high correlation[30]. After finding highly correlated features, we have to select one feature among the two based on certain conditions.
- *Selection of Features using CFS subset Evaluator (CFS):* This technique assesses the effectiveness of the subset of features by taking into consideration the predictive ability of each feature. This technique selects the subset of features with low inter-correlation but is highly correlated with the target class[31].
- *Selection of features using genetic algorithm (GA)* Genetic algorithm[32] helps to search for the best set of features that can improve the performance of the models. The advantage of this technique is that it permits the best solution to rise out of the best of earlier solutions. The core idea of this technique is to combine the various solutions from generation to extract the best features(genes) from each one to create new and more fitted individuals. Figure 2 shows the flowchart for GA to find the best sets of metrics. Initially, we generated 50 numbers of chromosomes with each gene of the chromosome containing the value 0 or 1, i.e., 0 for not considering features and 1 for considering features. Then, we computed the fitness value of each chromosome using Equation 5. Equation 5 is designed to maximize the accuracy and minimize the number of features. After finding the fitness value of all chromosomes, we selected the chromosome with a higher fitness value. The higher fitness value chromosome compared with the stopping condition. If satisfied, stop; otherwise, we will proceed with the next step. The next step is to apply crossover and mutation to all chromosomes and get half the number of the chromosomes i.e., two chromosomes combined using crossover to get one chromosome. The remaining half of the chromosomes are generated randomly. The above process will continue until we meet the stopping conditions.

$$Fitness = 0.8 * Accuracy + 0.2 * \frac{Total_{Features} - Selected_{Features}}{Total_{Features}} \quad (5)$$

Classification techniques

The primary objective of this research is to find the pattern based on source code metrics extracted from the WSDL file's Java file that help to predict anti-patterns present in unseen WSDL files. These patterns are identified using thirty different variants of machine learning techniques as shown in Table 4. These machine learning are validated using 5-fold cross-validation, and their ability to predict anti-patterns is computed in terms of accuracy and AUC values.

Proposed framework

Figure 3 shows our framework consisting of several steps. The dataset contemplated has a set of WSDL files considered as the input. The detailed steps of the proposed framework are given below:

- As shown in Fig. 3, we have calculated the Chidamber and Kemerer Java Metrics(CKJM) for each Java file generated from the WSDL file. Then, we applied different aggregation measures to the CKJM metrics computed from each Java file to generate file-level metrics.
- After finding metrics at the system level using different aggregation measures, we have also applied feature selection techniques to find the relevant set of features and remove irrelevant features. This set of metrics is later used as input to generate models for detecting web service anti-patterns. The Min-max normalization approach is used for normalizing the values of all selected features in the range of 0 to 1.

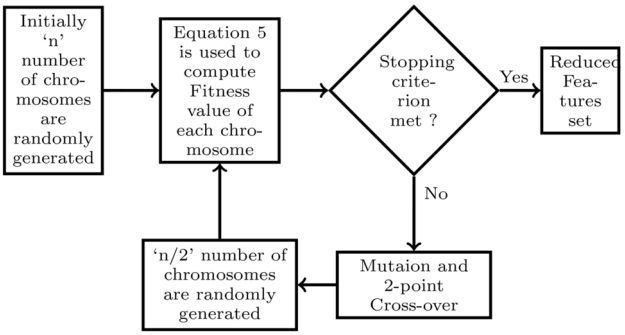


Fig. 2. Flow chart representing GA execution.

Classifiers	Description
Naive Bayes Algorithm (NB)[33]	Notable for multi-class prediction. Utilizing this algorithm, we can foresee the likelihood of different classes of target variables. In this work, we use three variants of naive Bayes algorithms to generate models for predicting web service anti-patterns i.e., Gaussian Naive Bayes(GNB), Multinomial Naive Bayes(MNB), Bernoulli Naive Bayes(BNB).
Decision Tree (DT)[34]	Represents the estimate of a target variable via the use of several independent variables in a decision model.
Logistic Regression Analysis (LOGR)[35]	A statistical approach used to analyze a dataset in which there are one or more independent variables that may be used to predict the outcomes of a dependent variable
Support Vector Classifier (SVC)[36]	It functions as a non-probabilistic binary linear classifier by classifying input data into one of two categories which makes it an excellent choice for developing a classification model. SVC with three different kernels i.e., linear (SVC-L), polynomial (SVC-P), and radial (SVC-R) are employed for training models in this work.
Least Square Support Vector Machine (LSSVM)[37]	this algorithm applies minimization of the sum of squared errors to the objective functions. This is a supervised learning method that analyzes data to recognize patterns. LSSVM with linear (LSSVM-Lin), Polynomial (LSSVM-Poly), and Radial Basis Functions (LSSVM-RBF) are used for training the models.
Extreme Learning Machine (ELM)[38]	This is a learning procedure for single hidden layer feed-forward neural networks. The key component of this approach is the random creation of hidden nodes, in which hidden node parameters are assigned at random, regardless of training samples. The anti-pattern detection models were trained with ELM using linear (ELM-Lin), polynomial (ELM-Poly), and radial basis functions (ELM-RBF).
Weighted Extreme Learning Machine (WELM)[39]	When dealing with imbalanced data, this approach gives more weight to the minority class and less weight to the majority class. WELM selects a weighting scheme based on the class distribution, and the weights created are inversely proportional to the number of samples in the training set. We implemented four different kernel functions (Sigmoid, Radbas, Tribes, and Sine) to WELM to boost its speed even further.
Multi-Layer Perceptron (MLP)[40]	MLP can train a non-linear function approximator for either classification or regression from a collection of features and a target. It is different from logistic regression because there can be one or more non-linear layers, called hidden layers, between the input and output layers.
MLP with Stochastic Gradient Descent (MLP-SGD)	It is necessary to update the weights to reduce output error while using MLP. SGD is employed for this purpose. The SGD technique finds the minima in error space by taking the 1st-order derivative of the total error function.
MLP with Quasi-Newton Method (MLP-LNF)	is a quick optimization approach that may be used as an alternative to conjugate gradient methods. Calculating the 2nd order derivatives of the total error function for each component of the gradient vector is required for this technique to be effective.
MLP with Stochastic Gradient with Adaptive Learning Rate Method (MLP-ADAM)	As the sample size is too small, the training procedure will take excessive time to converge. Although it is theoretically feasible to predict the best value of the learning rate (α) before training, it is practically impossible to predict the value of changes throughout the training process. Thus, ADAM is employed for training the prediction model in this study.
K-Nearest Neighbour (KNN)[41]	KNN is a non-parametric algorithm, which implies that it makes no assumptions about the data it is given as an input. It is sometimes referred to as a lazy learner algorithm since it does not learn from the training set immediately; instead, it stores the dataset and then acts on the dataset when it comes time to classify the data. During the training phase, the KNN algorithm saves the dataset and then classifies new data into a category that is very comparable to the latest data.
Bagging Classifier (BAG)[42]	is an ensemble meta-assessor that fits base classifiers each on subjective subsets of the underlying dataset and afterward aggregates their remote predictions performed either via voting or using averaging to form the concluding prediction.
Random Forest Classifier (RF)[43]	This algorithm makes decision trees on data samples and a while later receives the prediction from all of them and finally chooses the best arrangement using the method of voting.
Extra Trees Classifier (EXTR)[44]	This actualizes a meta-assessor that suits different randomized decision trees or extra-trees on different sub-samples of the dataset and utilizes averaging to enhance the predictive accuracy and supervises over-fitting.
AdaBoost Classifier (AdaB)[45]	is a meta-estimator that starts evolving by fitting a classifier on the first dataset and later on fits more duplicates of the classifier on the equivalent dataset; however, the weights of incorrectly classified instances are changed with the end goal ensuing classifiers revolve more around troublesome cases.
Gradient Boosting Classifier (GraB)[46]	The ideology of the GraB classifier is to restrict the loss or the differentiation between the actual class estimation of the training instance and the predicted class esteem. It facilitates constructing an additive model in a forward stage-wise style.
Deep Learning Technique (DL)[47]	Deep learning uses artificial neural networks, a kind of machine learning that works dependent on the structure and capacity of the human brain. This algorithm uses various instances from the dataset or relevant examples for training the machines. The primary benefit of an ANN over other types of algorithms is its novel information processing architecture. In this work, we have used Deep Learning(DL) technique with a distinct number of hidden layers, i.e., DL with one hidden layer(DL1), DL with two hidden layers (DL2), DL with three hidden layers (DL3), DL with four hidden layers (DL4), DL with five hidden layers (DL5) and DL with six hidden layers (DL6).

Table 4. Classification technique.

- While reviewing and inspecting the datasets, we observed that the considered data have an imbalanced nature of classes. Henceforth, to handle the class imbalance problem and its impact on the prediction accuracy of the models, we have also used five data sampling techniques. We compare the performance of the models generated using this sampling technique with the model developed using the original data (ORGD).
- After finding the balanced data with relevant sets of features as shown in Fig. 3, we have used a wide variety of classifiers. These techniques comprise general ML classifiers (LOGR, DT, etc..), Advanced deep learning classifiers (ELM, WELM, etc..), DL with distinct hidden layers (DL1, DL2, etc..), and Ensemble classifiers (BAG, EXTRA, etc..) to train the anti-pattern models and find important patterns that help to identify anti-pattern on future data. These models are validated using a 5-fold cross-validation approach. Table 17 contains the hyper-parameters used for model development.
- Finally, the impact and dependability of these techniques are measured using different performance parameters such as AUC and Accuracy. Table 5 shows the naming conventions used in this work.

Results and analysis

In this segment of the paper, we showcase the results & performance obtained from feature ranking and feature subset selection techniques over class-level metrics on the imbalanced and balanced dataset generated from sampling techniques. To get these balanced datasets, we first used the stated five different sampling techniques to overcome the class imbalance issue. Then we employed different variants of classifiers to detect the detect anti-patterns. The model's effectiveness was computed using different performance parameters. Considering the

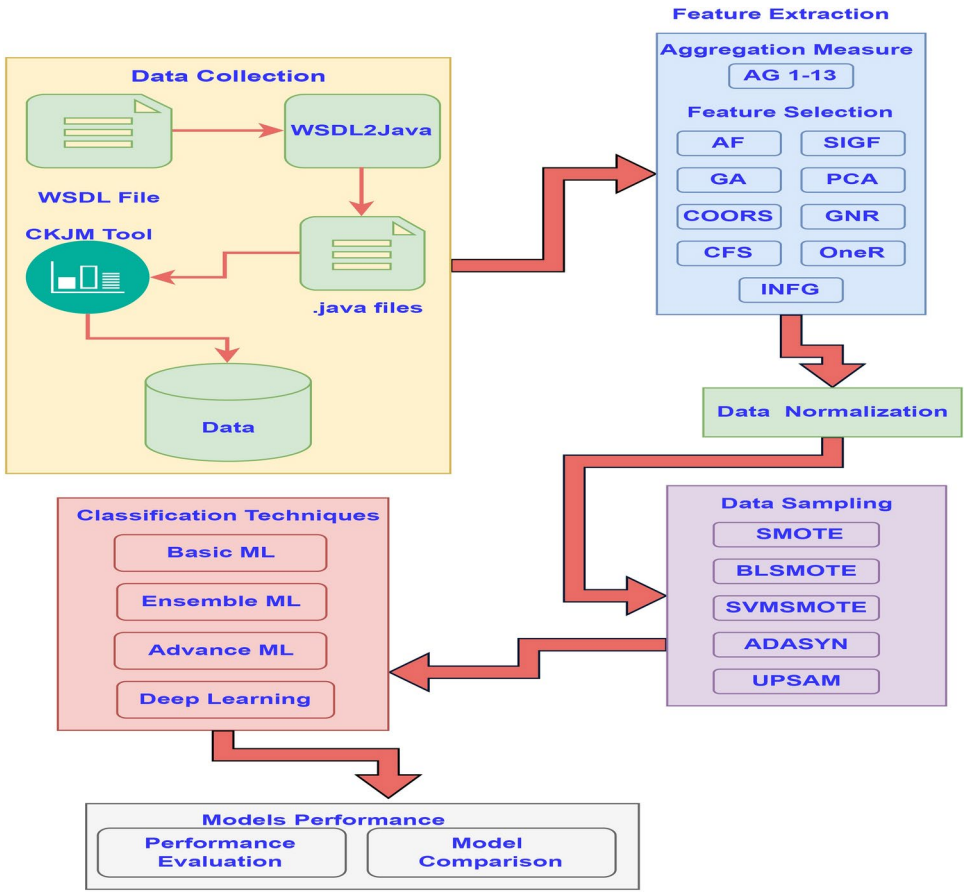


Fig. 3. Research framework for anti-pattern prediction.

space constraint, we have included the results of the randomly selected one-feature ranking technique and one-feature subset selection technique.

Feature selection results

Here, in this study, we would like to compare and contrast feature-subset selection and feature ranking techniques to examine if any of the techniques is superior to the others or if all the techniques perform equally well.

Relevant feature sets are generated after the application of the feature selection techniques, namely: Significant Features(SIGF) obtained by applying the Wilcoxon sign test, Information Gain(INFG), Gain Ratio(GNR), Correlation coefficient(CORR), Genetic Algorithm(GA), CFS subset evaluator(CFS), OneR, Principal Component Analysis(PCA) along with the 13 aggregation techniques namely: variance, arithmetic mean, skewness, median, quartile1, theli index, standard deviation, quartile3, generalized entropy, maximum, gini index, kurtosis and atkinson index are used as input for the generation of models for the detection of web service anti-patterns. Along with this, a model using the original dataset(OD) is also generated for detecting web service anti-patterns. The sets of features selected after applying each of the feature selection techniques considered are given in Tables 6, 7, 8, 9 and 10. Tables 6, 7, 8, 9 and 10 contains the results for anti-pattern type 1 to 5. The information present in Tables 6 suggested that the features like Q1(WMC), Mean(CBO), Gini index(CBO), Hoover index(CBO), Generalized entropy (RFC), skewness(LCOM), Q1(Ca), Max(MOA) are best set of features identified using information gain for AP1.

Accuracy and AUC values analysis

In this work, We used a wide range of classifier techniques to find the important pattern that helps identify different types of anti-patterns in web service. Initially, we have tried with most frequently used classifier techniques such as three variants of Naive Bayes, Support Vector Classifier with linear Kernel (SVC-LIN), SVC with the polynomial kernel (SVC-POLY), SVC with radial bias kernel (SVC-RBF), Logistic Regression Analysis (LOGR) to find an important pattern. After, we used the advanced level of machine learning like extreme learning machine(ELM), Least square SVM, weighted extreme learning machine (WELM) with different kernels, and Ensemble classifiers such as AdaBoost Classifier(AdaB), Random Forest Classifier(RF), Bagging Classifier(BAG), Extra Trees Classifier(EXTR), and Gradient Boosting Classifier(GraB). Further, the deep layer technique with a varying number of hidden layers has also been used to find the important pattern that helps to identify different types of anti-patterns present in web services. These techniques are validated using 5-fold cross-validation approaches and compared using Accuracy and AUC performance values on the testing data.

Abbreviation	Corresponding name	Abbreviation	Corresponding name
AG1	Variance	SIGF	Significant Features
AG2	Arithmetic Mean	INFG	Information Gain Attribute Ranking
AG3	Skewness	GNR	Gain Ratio Ranking
AG4	Minimum	OneR	OneR attribute evaluation
AG5	Median	PCA	Principal Component Analysis
AG6	Quartile1(25%)	CORR	Correlation Coefficient Analysis
AG7	Theli Index	CFS	Classifier subset Evaluator
AG8	Standard Deviation	GA	Genetic Algorithm
AG9	Quartile3(75%)	ELM	Extreme learning machine
AG10	Generalized Entropy	WELM	Weighted extreme learning machine
AG11	Maximum	SVC-LIN	Support Vector Classifier with linear Kernel
AG12	Gini Index	DL	Deep Learning
AG13	kurtosis	MLP-ADA	MLP with stochastic gradient-based optimizer proposed by Kingma
AG14	Hoover Index	DL-1	Deep Learning with 1 hidden Layer
AG15	Atkinson Index	GraB	Gradient Boosting Classifier
AG16	Shannon Entropy	EXTR	Extra Trees Classifier
AdaB	AdaBoost Classifier	FGWS	Fine-Grained anti-pattern
CSW	Chatty anti-pattern	RF	Random Forest Classifier
GOWS	God Object anti-pattern	BAG	Bagging Classifier
DWS	Data anti-pattern	MLP-SG	MLP with stochastic gradient descent.
AWS	Ambiguous Anti-pattern	LSSVM	Least square SVM
GNB	Gaussian Naive Bayes	ADASYN	Adaptive Synthetic Sampling Technique
BNB	Bernoulli Naive Bayes	SMOTE	Synthetic Minority Oversampling Technique
MNB	Multinomial Naive Bayes	SVMSMOTE	Support Vector Machine SMOTE
BLSMOTE	Borderline SMOTE	SVC-POLY	SVC with the polynomial kernel
UPSAM	UP sampling Technique	SVC-RBF	SVC with radial bias kernel
KDD	Knowledge Data Discovery	LOGR	Logistic Regression Analysis
FS	Feature Selection	MLP	Multi-layer Perceptron classifier
DT	Decision Tree	MLP-LNF	MLP with quasi-Newton methods

Table 5. Naming conventions.

AP1	AP2	AP3	AP4	AP5
Q1(wmc)	Gini index(cbo)	Gini index(dit)	Q1(wmc)	skewness(rfc)
Mean(cbo)	Hoover index(cbo)	Atkinson index(dit)	Q3(wmc)	Hoover index(rfc)
Gini index(cbo)	Q1(ce)	Shannon entropy(dit)	Median(noc)	Atkinson index(rfc)
Hoover index(cbo)	kurtosis(dam)	Min(noc)	Std(noc)	Shannon entropy(rfc)
Generalized entropy(rfc)	skewness(dam)	Max(noc)	Q1(noc)	Generalized entropy(rfc)
skewness(lcom)	Min(moa)	Median(noc)	Generalized entropy(ca)	Theil index(rfc)
Q1(ca)	Mean(moa)	Std(noc)	Theil index(ca)	Min(lcom)
Max(moa)	kurtosis(cam)	Q1(noc)	Std(cam)	skewness(cam)

Table 6. Features selected after applying information gain for all the anti-patterns.

In this work, we have also examined the benefit of using different variants of sampling techniques like SMOTE, UPSAMPLING, BLSMOTE, etc., to handle the class-imbalanced nature of data sets. To deal with the feature redundancy problem, we have used different variants of aggregation techniques to find system-level metrics using class-level metrics without losing important information. Further, different variants of feature selection techniques have also been used to remove irrelevant metrics and find the best combination of reverent metrics. Tables 11, 12, and 13 show the accuracy and AUC values of the models trained using the most frequently used classifiers, advanced level of classifiers, and ensemble learning. The rows of the tables are used to represent the input metrics for the models, and columns are used to represent the classifiers used to train the models, i.e., the trained anti-pattern prediction model using MNB by taking all features as an input achieved 84.96% of Accuracy and 0.86 value of AUC. The AUC value greater than 0.7 confirms that the trained models have the ability to predict anti-patterns using source code metrics. The high-value AUC in the case of advanced level of machine learning confirms that the models trained using the advanced level of machine learning, like LSSVM with different kernels, and WELM with different kernels, have better ability for anti-pattern prediction as compared

AP1	AP2	AP3	AP4	AP5
Var(noc)	kurtosis(noc)	Atkinson index(dit)	Min(noc)	skewness(rfc)
Theil index(cbo)	Var(cbo)	Min(noc)	Median(noc)	Hoover index(rfc)
Generalized entropy(rfc)	Theil index(rfc)	Max(noc)	Std(noc)	Atkinson index(rfc)
Min(lcom)	Std(lcom)	Median(noc)	Q1(noc)	Shannon entropy(rfc)
Max(lcom)	Atkinson index(lcom)	Std(noc)	kurtosis(noc)	Generalized entropy(rfc)
Std(lcom)	Gini index(ca)	Var(noc)	Atkinson index(lcom)	Theil index(rfc)
skewness(lcom)	Hoover index(ca)	Q1(noc)	Theil index(lcom)	Min(lcom)
Median(cam)	Shannon entropy(npm)	Min(loc)	Min(loc)	skewness(cam)

Table 7. Features selected after applying gain ratio for all the anti-patterns.

AP1	AP2	AP3	AP4	AP5
Hoover index(cbo)	Atkinson index(lcom)	Gini index(dit)	Gini index(dit)	Std(wmc)
kurtosis(rfc)	Q1(ce)	Atkinson index(dit)	Atkinson index(dit)	kurtosis(cbo)
Generalized entropy(rfc)	kurtosis(ce)	Shannon entropy(dit)	Min(noc)	Generalized entropy(cbo)
Min(lcom)	Gini index(ce)	Max(noc)	Max(noc)	Var(rfc)
Max(lcom)	Hoover index(lcom3)	Min(lcom)	Min(lcom)	Atkinson index(ca)
Std(lcom)	kurtosis(dam)	Median(cam)	Max(lcom)	Q1(npm)
Var(lcom)	skewness(dam)	Std(cam)	Theil index(lcom)	Q3(loc)
Mean(ca)	Generalized entropy(dam)		Std(cam)	Theil index(dam)

Table 8. Features selected after applying correlation coefficient for all the anti-patterns.

AP1	AP2	AP3	AP4	AP5
Var(noc)	Var(cbo)	Std(noc)	Q3(wmc)	Min(wmc)
Q3(noc)	Std(lcom)	Var(noc)	Atkinson index(dit)	Std(wmc)
Gini index(cbo)	Gini index(ca)	Q3(rfc)	Median(noc)	Var(wmc)
Generalized entropy(rfc)	Hoover index(ca)	skewness(rfc)	Std(noc)	
Min(lcom)	Q1(ce)	Q3(lcom)	Q1(noc)	
Std(lcom)	kurtosis(dam)	Atkinson index(lcom)	kurtosis(noc)	
Std(cam)		Q1(ce)	Min(lcom)	
		Min(loc)	Q3(lcom)	
		Shannon entropy(dam)	Atkinson index(lcom)	
		Mean(cam)	Theil index(lcom)	

Table 9. Features selected after applying CFS subset evaluator for all the anti-patterns.

AP1	AP2	AP3	AP4	AP5
Shannon entropy(noc)	Gini index(rfc)	Q3(wmc)	Q1(wmc)	Median(lcom)
Generalized entropy(rfc)	Hoover index(rfc)	Median(noc)	Median(noc)	Var(lcom)
Min(lcom)	Atkinson index(rfc)	Std(noc)	Q3(noc)	Q1(lcom)
Max(lcom)	Shannon entropy(rfc)	Q1(noc)	Std(cbo)	Q3(lcom)
Std(lcom)	Generalized entropy(rfc)	skewness(noc)	Q3(rfc)	skewness(lcom)
Median(ce)	Theil index(rfc)	Gini index(noc)	kurtosis(rfc)	Hoover index(lcom)
Max(moa)	Q1(lcom)	Theil index(ca)	Hoover index(lcom)	Atkinson index(lcom)
Median(cam)	skewness(cam)	Shannon entropy(mfa)	Theil index(lcom)	skewness(cam)

Table 10. Features selected after applying OneR for all the anti-patterns.

to other techniques. Similarly, the models trained on sampled data have a better ability to predict as compared to the original data. Finally, the models developed by taking selected sets of features as input have a higher value of AUC. Accuracy confirms that the models trained on reduced sets of features have a better capability of anti-pattern prediction than all features.

	Accuracy									AUC								
	MNB	BNB	GNB	DT	LOGR	KNN	SVL	SVP	SVR	MNB	BNB	GNB	DT	LOGR	KNN	SVL	SVP	SVR
ORG-DATA																		
OD	84.96	69.03	68.58	90.27	**92.92**	91.59	88.94	90.27	89.82	0.86	0.71	0.76	0.70	**0.88**	0.80	0.83	0.85	0.85
SIGF	86.28	69.91	84.07	88.05	**92.48**	92.04	89.82	89.82	88.94	0.86	0.72	0.82	0.67	**0.89**	0.80	0.88	0.87	0.88
AG1	90.27	76.99	87.61	91.15	91.15	**92.04**	88.94	89.38	88.05	0.51	0.72	**0.89**	0.72	0.84	0.75	0.86	0.71	0.77
AG2	**90.71**	87.61	68.58	86.28	**90.71**	89.38	66.81	80.09	71.24	0.71	0.68	**0.80**	0.57	0.77	0.71	0.73	0.78	0.78
AG3	89.82	86.28	85.40	88.05	**90.71**	90.27	86.73	88.50	87.17	0.85	0.67	**0.89**	0.65	0.87	0.75	**0.89**	0.74	0.82
AG4	89.82	86.28	73.45	89.82	90.27	**91.15**	87.17	90.27	88.05	0.85	0.67	0.84	0.66	0.84	0.70	**0.89**	0.82	0.84
AG5	**90.71**	90.27	57.96	89.38	**90.71**	89.38	55.75	58.41	56.19	0.56	0.57	0.69	0.63	0.71	0.70	**0.73**	0.72	0.71
AG6	**90.71**	**90.71**	71.68	85.40	**90.71**	88.05	47.79	51.77	59.73	0.60	0.62	**0.70**	0.49	0.63	0.60	0.62	0.61	0.67
AG7	90.27	87.17	91.15	90.27	90.71	**92.04**	90.71	89.82	89.82	0.82	0.67	**0.88**	0.68	0.87	0.78	0.84	0.51	0.84
AG8	90.27	87.17	90.27	89.38	90.71	**91.15**	88.05	88.50	89.38	0.87	0.67	0.86	0.70	**0.88**	0.75	0.87	0.77	0.86
AG9	**90.71**	89.82	80.09	88.05	**90.71**	87.61	69.47	74.78	73.45	0.70	0.35	**0.84**	0.61	0.74	0.73	0.72	0.63	0.72
AG10	**90.71**	83.19	78.32	86.28	**90.71**	88.94	71.24	82.74	79.65	0.74	0.68	**0.83**	0.61	0.80	0.76	0.81	0.77	0.81
AG11	**90.71**	82.74	83.63	85.40	**90.71**	88.94	76.99	86.28	80.53	0.78	0.68	0.84	0.57	**0.85**	0.81	0.84	0.76	0.82
AG12	90.71	90.27	89.38	90.27	91.15	**92.48**	91.15	88.05	89.82	0.85	0.65	0.82	0.73	0.84	0.75	0.83	0.83	**0.87**
AG13	**90.71**	82.74	84.07	87.17	**90.71**	88.50	76.99	86.73	81.86	0.79	0.68	**0.85**	0.60	**0.85**	0.80	0.83	0.68	0.82
INFG	89.82	72.57	76.11	90.71	**92.04**	90.27	88.05	89.82	88.50	0.87	0.72	0.83	0.71	**0.88**	0.76	0.86	**0.88**	0.87
GNR	91.15	89.82	92.48	91.59	91.15	**93.81**	89.38	89.38	88.94	0.91	0.39	**0.92**	0.71	0.91	0.86	0.90	0.90	0.91
CORR	91.15	89.82	91.59	91.59	91.59	91.15	**92.04**	90.71	**92.04**	0.86	0.42	0.86	0.71	0.85	0.86	0.86	0.86	**0.87**
CFS	90.71	90.27	90.27	92.04	92.48	**93.81**	87.17	91.15	88.94	0.74	0.38	**0.92**	0.74	0.88	0.86	0.89	0.65	0.86
OneR	90.71	90.71	**92.48**	92.04	91.59	90.27	88.94	89.82	88.50	**0.90**	0.42	**0.90**	0.76	0.87	0.81	0.88	0.87	0.89
GA	91.15	89.82	91.59	**92.48**	90.71	91.15	90.27	90.27	89.82	**0.90**	0.38	0.89	0.74	0.86	0.86	0.86	0.85	**0.90**
PCA	90.71	90.27	88.94	89.38	90.71	89.38	**92.04**	90.27	89.82	0.48	0.51	0.76	0.68	0.67	0.81	**0.90**	0.87	0.82
SMOTE-DATA																		
OD	80.98	82.93	82.44	**93.17**	91.22	90.24	75.61	76.83	76.34	0.89	0.84	0.85	0.93	**0.96**	**0.96**	0.65	0.65	0.65
SIGF	79.76	82.93	87.07	**93.41**	90.24	91.95	76.83	78.05	77.56	0.89	0.84	0.88	0.93	0.96	**0.97**	0.64	0.65	0.65
AG1	57.56	83.17	82.68	**91.95**	85.37	88.78	63.90	62.20	67.80	0.71	0.85	0.91	0.92	0.92	**0.95**	0.53	0.53	0.53
AG2	73.66	82.93	77.32	**90.73**	76.10	86.34	68.05	73.41	72.44	0.78	0.84	0.88	0.91	0.81	**0.93**	0.55	0.59	0.58
AG3	80.00	83.41	83.90	89.02	86.34	**91.95**	81.22	84.63	83.90	0.88	0.84	0.91	0.89	0.92	**0.96**	0.76	0.77	0.77
AG4	76.83	83.17	77.32	**91.95**	81.46	90.98	71.46	71.46	74.63	0.87	0.85	0.89	0.92	0.89	**0.96**	0.60	0.62	0.62
AG5	67.80	66.34	72.93	**84.88**	73.17	83.41	67.32	67.80	68.05	0.72	0.62	0.79	0.89	0.76	**0.91**	0.53	0.55	0.55
AG6	65.12	71.71	71.22	**86.59**	68.54	83.41	60.98	62.93	63.66	0.67	0.68	0.77	0.86	0.68	**0.87**	0.49	0.50	0.51
AG7	83.90	82.20	80.73	**91.95**	84.15	**91.95**	72.68	71.71	76.10	0.89	0.84	0.90	0.92	0.93	**0.97**	0.64	0.64	0.65
AG8	81.46	82.44	84.39	91.71	86.10	**91.95**	66.59	69.02	68.29	0.91	0.83	0.92	0.92	0.93	**0.96**	0.53	0.54	0.53
AG9	78.29	48.05	82.68	**93.17**	78.29	83.90	61.46	66.10	64.63	0.81	0.47	0.92	**0.93**	0.83	**0.93**	0.51	0.53	0.53
AG10	78.29	79.02	82.93	85.37	81.46	**87.07**	72.20	73.17	74.15	0.82	0.78	0.89	0.85	0.87	**0.93**	0.60	0.62	0.62
AG11	78.78	79.02	82.93	87.80	84.63	**89.02**	74.15	76.34	76.10	0.85	0.79	0.90	0.88	0.90	**0.94**	0.58	0.59	0.60
AG12	75.85	82.20	81.46	**91.22**	79.51	86.83	68.54	73.41	71.22	0.89	0.80	0.87	0.91	0.92	**0.94**	0.61	0.61	0.61
AG13	79.02	79.02	83.17	**88.78**	84.63	88.05	65.61	67.07	66.83	0.86	0.80	0.90	0.89	0.91	**0.94**	0.53	0.54	0.54
INFG	81.22	82.20	83.41	**91.71**	89.27	89.76	70.49	70.98	70.98	0.89	0.87	0.87	0.92	0.94	**0.96**	0.52	0.52	0.52
GNR	85.37	56.59	84.63	**87.8**	86.10	87.07	73.17	73.90	73.41	0.92	0.52	0.92	0.88	**0.94**	0.93	0.62	0.61	0.61
CORR	85.37	56.34	82.93	89.27	84.15	**91.22**	65.12	65.61	65.37	0.92	0.49	0.90	0.90	0.91	**0.95**	0.50	0.51	0.50
CFS	67.32	57.07	83.41	**88.54**	83.90	86.59	78.29	80.00	81.71	0.78	0.48	0.91	0.88	0.91	**0.93**	0.69	0.68	0.69
OneR	85.61	56.34	84.63	89.27	86.83	**91.46**	67.80	67.80	67.56	0.93	0.53	0.92	0.90	0.93	**0.96**	0.52	0.54	0.52
GA	86.34	56.59	83.66	89.27	85.85	**91.46**	66.59	66.83	67.07	0.94	0.52	0.92	0.89	0.93	**0.95**	0.53	0.54	0.53
PCA	66.59	61.71	81.95	**91.95**	75.37	89.27	64.15	72.93	70.24	0.65	0.59	0.85	0.92	0.87	**0.95**	0.61	0.62	0.61
BLSMOTE-DATA																		
OD	83.41	83.41	84.15	**94.88**	92.44	88.78	69.51	69.51	68.54	0.90	0.86	0.86	0.95	**0.96**	0.95	0.53	0.53	0.53
SIGF	82.20	83.41	90.73	91.71	**92.44**	88.78	77.32	77.32	78.05	0.89	0.86	0.91	0.92	0.95	**0.96**	0.64	0.65	0.65
AG1	59.76	83.41	84.88	**93.66**	86.59	91.71	77.80	72.44	78.54	0.61	0.87	0.93	0.94	0.93	**0.97**	0.58	0.57	0.58
AG2	77.32	82.93	88.29	**94.63**	84.39	91.22	72.93	78.54	77.07	0.81	0.82	0.95	0.94	0.85	**0.96**	0.56	0.61	0.60
AG3	83.17	83.66	89.27	**92.20**	90.00	91.46	69.76	68.05	69.27	0.92	0.84	0.96	0.92	0.95	**0.97**	0.53	0.53	0.53
AG4	84.63	83.66	87.56	92.20	88.78	**92.68**	69.76	69.76	70.73	0.93	0.85	0.95	0.93	0.94	**0.96**	0.51	0.52	0.51
AG5	67.07	66.34	74.88	84.63	75.85	**84.88**	72.44	73.17	72.68	0.67	0.58	0.78	0.88	0.77	**0.90**	0.52	0.54	0.53
AG6	70.73	73.41	73.90	**84.88**	72.93	81.22	65.37	68.05	67.80	0.73	0.72	0.82	0.84	0.74	**0.89**	0.55	0.58	0.59

Continued

	Accuracy									AUC								
	MNB	BNB	GNB	DT	LOGR	KNN	SVL	SVP	SVR	MNB	BNB	GNB	DT	LOGR	KNN	SVL	SVP	SVR
AG7	80.98	78.78	81.46	90.49	85.85	**91.46**	73.66	71.95	76.34	0.88	0.80	0.90	0.91	0.91	**0.96**	0.63	0.63	0.64
AG8	81.95	81.71	86.59	**92.68**	86.34	92.20	67.07	67.80	68.05	0.86	0.81	0.89	0.93	0.90	**0.96**	0.54	0.55	0.55
AG9	83.41	50.49	89.76	**95.37**	83.41	88.54	72.20	76.10	76.10	0.84	0.46	0.94	**0.95**	0.86	**0.95**	0.59	0.62	0.62
AG10	80.49	80.98	85.37	**89.02**	84.63	86.59	59.51	59.76	60.00	0.81	0.79	0.91	0.89	0.89	**0.95**	0.50	0.50	0.50
AG11	80.98	81.22	85.85	**89.27**	86.10	89.02	81.46	83.90	83.17	0.87	0.81	0.92	0.89	0.93	**0.94**	0.76	0.77	0.77
AG12	79.76	82.44	84.39	**91.22**	82.68	88.54	72.68	74.63	72.44	0.91	0.80	0.90	0.91	0.92	**0.95**	0.62	0.62	0.63
AG13	79.27	79.27	82.68	**88.05**	84.39	86.34	66.83	67.07	67.32	0.84	0.77	0.90	0.88	0.91	**0.94**	0.52	0.52	0.52
INFG	83.17	82.68	85.37	**93.17**	90.49	89.02	78.54	79.27	78.29	0.90	0.88	0.89	0.94	**0.96**	**0.96**	0.60	0.61	0.61
GNR	90.24	56.59	86.34	91.95	87.80	**92.68**	67.32	68.29	68.29	0.94	0.53	0.94	0.92	0.95	**0.97**	0.54	0.54	0.54
CORR	86.59	56.34	84.63	**92.93**	86.34	91.95	66.59	68.05	66.34	0.93	0.50	0.91	0.93	0.92	**0.96**	0.52	0.52	0.52
CFS	73.41	57.07	79.02	**90.73**	80.73	87.80	65.61	65.85	67.80	0.77	0.51	0.90	0.91	0.89	**0.93**	0.51	0.50	0.51
OneR	84.63	56.34	84.15	91.71	85.85	**92.20**	74.63	76.59	75.37	0.94	0.51	0.93	0.92	0.94	**0.97**	0.62	0.62	0.62
GA	88.05	56.59	86.34	**94.63**	87.07	94.15	74.39	77.56	76.10	0.94	0.53	0.94	0.94	0.93	**0.96**	0.64	0.66	0.64
PCA	66.59	61.71	56.83	**90.49**	80.98	88.54	67.56	72.44	70.24	0.65	0.57	0.78	0.91	0.90	**0.95**	0.61	0.62	0.61

Table 11. Accuracy and AUC for Anti-pattern 1: Most Frequently Used Classifiers. Best performance value in bold.

RQ 1:	Can web-service anti-patter prediction models be developed using source code metrics and machine learning?
ANS:	The high value of AUC, i.e., greater than 0.7, as shown in Table 11, 12, and 13 confirms that the developed models have the ability to predict anti-patterns based on source code metrics. The experimental findings confirmed that the models performed better after applying sampling and FS techniques.

Comparative analysis

This research aims to evaluate the impact of feature selection techniques, data-sampling techniques, and a wide variety of machine learning on the performance of the web-service anti-pattern prediction models. Considering this, we have applied twenty-two different sets of features, five different data-sampling, and thirty-two different classifiers for anti-pattern prediction models. The predictive power of these techniques is computed using Accuracy & AUC and compared with the help of box-plot diagrams and hypothesis rank-sum techniques. The final intensive assessment and performance of these techniques individually are presented in subsequent subsections.

Aggregation measures and feature selection techniques

In our experiment, different aggregation measures were used to find the source code metrics at the system level from the class level without losing information. Further, eight feature selection techniques have also been used to remove irrelevant and redundant features. After applying aggregation measures and feature selection techniques along with the original features, all these feature sets are used as input for developing the models for detecting web service anti-patterns. Finally, Statistical and AUC studies were used to determine the significance and reliability of various feature selection strategies on five different types of anti-patterns.

Comparison of different aggregation measures and sets of features: Descriptive statistics and box-plot The Fig. 4a, b of Fig. 4 depict the box-plot for the Accuracy and AUC of different aggregation measures and sets of features. The descriptive statistics of all employed feature selection techniques are presented in Table 14. The following conclusions can be drawn from Fig. 4 and Table 14:

- All the models give reasonable accuracies ranging between 75-95 % and AUC values between 0.8-0.95.
- The models trained on all features achieves 83.35 mean accuracy and 0.80 as mean AUC.
- Among the aggregation measures, AG3 shows the best performance, with a mean AUC value of 0.86. At the same time, the model developed using the feature set computed by using AG6 as input shows the worst performance, with a mean AUC value of 0.76.
- Among all the feature sets which are considered as input for developing the models to detect web service anti-patterns, SIGF is the best model, with a mean AUC value of 0.88. In contrast, the model developed with features selected by PCA as input is the worst model, with a mean AUC value of 0.71. The model developed by AG3 has the second-best performance, with a mean accuracy of 0.86.

Comparison of different aggregation measures and sets of features: Wilcoxon Signed Rank Test (WSRT) with Friedman mean rank (FMR): In this experiment, we have also employed two statistical tests for hypothesis analysis: Wilcoxon Signed Rank Test and Friedman Test. Initially, we applied WSRT to find pair-wise significant differences between the predictive capability of the models trained by taking different sets of features as input.

	Accuracy										AUC									
	MLPL	MLPS	MLPA	WELS	WELSI	WELR	WELT	LSVL	LSVP	LSVR	MLPL	MLPS	MLPA	WELS	WELSI	WELR	WELT	LSVL	LSVP	LSVR
ORG-DATA																				
OD	90.71	90.71	90.71	90.71	90.71	90.71	90.71	97.35	**98.67**	97.79	0.68	0.36	0.76	0.50	0.50	0.50	0.50	0.99	**1**	0.99
SIGF	92.04	90.27	91.15	84.07	86.73	87.17	83.63	96.02	**98.67**	98.23	0.78	0.85	0.84	0.86	0.84	0.87	0.85	0.98	**1**	**1**
AG1	92.04	90.71	92.04	69.91	77.88	75.66	73.01	95.58	**96.9**	94.69	0.76	0.59	0.85	0.86	0.86	0.85	0.83	0.94	**0.95**	0.93
AG2	88.94	90.71	90.71	66.81	68.58	68.58	66.37	92.48	91.15	**96.9**	0.66	0.32	0.75	0.80	0.79	0.81	0.79	0.88	0.93	**0.99**
AG3	88.50	90.71	90.71	73.89	80.97	75.66	79.65	94.25	**98.23**	**98.23**	0.69	0.55	0.86	0.83	0.85	0.84	0.82	0.94	**0.99**	**0.99**
AG4	88.50	90.71	91.59	73.45	76.11	77.88	74.34	92.92	96.46	**98.23**	0.71	0.47	0.85	0.82	0.86	0.83	0.84	0.93	0.96	**1**
AG5	90.71	90.71	90.71	43.36	52.21	52.65	52.65	90.71	91.15	**92.04**	0.69	0.22	0.27	0.73	0.76	0.76	0.74	0.78	0.80	**0.95**
AG6	**90.71**	**90.71**	**90.71**	53.54	55.31	58.41	53.98	**90.71**	**90.71**	**90.71**	0.64	0.40	0.42	0.65	0.69	0.71	0.70	0.74	**0.78**	0.74
AG7	90.71	90.71	91.15	89.38	89.82	89.38	87.61	93.36	**96.9**	94.25	0.36	0.77	0.90	0.89	0.90	0.90	0.90	0.88	**0.96**	0.92
AG8	90.71	90.71	90.27	83.63	86.73	85.84	86.28	94.25	**98.67**	95.58	0.36	0.61	0.76	0.89	0.89	0.89	0.89	0.90	**0.99**	0.98
AG9	90.71	90.71	90.71	72.12	71.68	69.03	68.14	90.71	90.71	**98.67**	0.78	0.47	0.66	0.78	0.77	0.78	0.75	0.82	0.87	**1**
AG10	86.28	90.71	90.71	70.80	72.12	73.89	73.45	90.71	94.25	**98.67**	0.68	0.60	0.76	0.79	0.80	0.80	0.79	0.89	0.95	**1**
AG11	87.17	90.71	91.15	71.68	76.99	73.45	71.68	90.71	**94.69**	**94.69**	0.76	0.69	0.84	0.81	0.83	0.84	0.84	0.89	**0.96**	**0.96**
AG12	91.59	90.71	91.15	92.04	89.82	89.82	90.71	94.69	**97.35**	93.36	0.79	0.67	0.84	0.87	0.83	0.83	0.87	0.90	**0.95**	0.91
AG13	88.05	90.71	90.71	72.57	72.57	76.11	73.45	90.71	**96.02**	93.36	0.70	0.69	0.84	0.84	0.84	0.83	0.83	0.88	0.97	**0.98**
INFG	90.71	90.71	90.71	90.71	90.71	90.71	90.71	96.90	**98.67**	**98.67**	0.36	0.36	0.36	0.50	0.50	0.50	0.50	0.98	**1**	**1**
GNR	91.59	90.71	92.04	92.04	92.04	91.59	91.15	93.36	97.35	**97.79**	0.89	0.32	0.87	0.92	0.90	0.91	0.91	0.90	0.97	**0.98**
CORRS	91.59	90.71	92.48	92.48	91.15	91.15	91.59	95.13	**97.35**	**97.35**	0.80	0.75	0.86	0.84	0.86	0.84	0.87	0.93	**0.98**	0.97
CFS	92.04	90.71	92.04	89.82	89.38	88.50	85.84	92.92	**96.9**	95.58	0.80	0.86	0.89	0.89	0.88	0.89	0.91	0.91	**0.95**	**0.95**
OneR	90.71	90.71	90.71	92.04	89.38	91.15	89.82	94.25	**95.13**	**95.13**	0.80	0.83	0.82	0.91	0.88	0.91	0.91	0.90	**0.95**	**0.95**
GA	90.27	90.71	90.71	90.71	91.15	91.59	91.15	95.13	**97.35**	95.13	0.85	0.85	0.86	0.88	0.87	0.90	0.89	0.94	**0.98**	0.93
PCA	88.94	90.71	90.71	92.04	90.71	88.05	89.38	91.59	95.13	**96.9**	0.84	0.68	0.75	0.81	0.80	0.75	0.71	0.90	0.92	**0.96**
SMOTE-DATA																				
OD	84.15	66.59	94.15	90.00	50.00	50.00	50.00	50.00	**97.8**	93.90	0.94	0.79	0.97	0.50	0.50	0.50	0.50	**1**	0.99	**1**
SIGF	95.37	93.90	96.10	89.76	85.61	93.17	90.24	87.32	**97.56**	94.63	0.96	0.97	0.98	0.93	0.97	0.96	0.95	**1**	0.99	**1**
AG1	93.17	87.07	91.95	89.27	74.88	77.80	78.78	81.46	88.54	**94.39**	0.97	0.93	0.96	0.86	0.90	0.91	0.93	0.94	0.98	**0.99**
AG2	91.46	77.07	**91.95**	86.34	74.15	75.85	73.90	75.37	82.44	90.00	0.95	0.82	0.95	0.80	0.81	0.82	0.82	0.89	0.97	**1**
AG3	94.63	84.88	95.61	89.27	80.24	83.17	83.17	83.41	89.51	**97.07**	0.95	0.91	0.97	0.88	0.91	0.90	0.91	0.95	**1**	**1**
AG4	94.88	81.71	93.90	89.27	77.07	78.05	75.85	78.54	86.59	**95.12**	0.96	0.91	0.97	0.85	0.88	0.86	0.88	0.94	0.99	**1**
AG5	77.80	69.76	72.20	**80**	68.54	69.27	69.51	71.22	76.83	78.05	0.80	0.74	0.75	0.72	0.74	0.73	0.74	0.78	0.81	**0.96**
AG6	66.34	50.49	69.51	**81.95**	66.10	69.51	65.85	65.85	70.73	78.05	0.69	0.45	0.69	0.68	0.71	0.69	0.70	0.71	0.83	**0.98**
AG7	**96.1**	84.63	93.90	91.71	82.44	83.17	84.39	85.37	84.63	93.17	0.97	0.92	0.98	0.91	0.92	0.93	0.94	0.93	0.98	**1**
AG8	93.90	85.37	94.39	90.49	83.90	85.37	83.66	84.15	90.98	**95.37**	0.96	0.92	0.97	0.92	0.93	0.92	0.92	0.96	0.99	**1**
AG9	46.59	50.00	71.22	**85.37**	78.54	78.29	77.32	78.05	78.29	83.90	0.46	0.47	0.83	0.82	0.81	0.82	0.81	0.84	0.91	**1**
AG10	**90.73**	78.54	84.88	84.88	77.80	78.54	78.05	79.27	84.88	**90.73**	0.92	0.85	0.92	0.83	0.85	0.86	0.87	0.91	0.96	**1**
AG11	90.24	84.88	88.78	87.07	78.54	80.73	78.05	80.24	85.12	**94.15**	0.94	0.91	0.94	0.85	0.90	0.87	0.86	0.92	0.98	**1**
AG12	87.07	68.29	84.63	87.07	74.63	77.80	78.29	78.29	84.88	**92.68**	0.93	0.73	0.92	0.89	0.91	0.87	0.84	0.92	**0.99**	**0.99**
AG13	**92.44**	85.12	90.00	84.88	78.78	82.20	81.95	81.22	85.85	91.71	0.95	0.91	0.95	0.86	0.89	0.88	0.88	0.92	0.97	**1**
INFG	44.63	89.76	93.41	88.29	50.00	50.00	50.00	50.00	97.32	**98.78**	0.43	0.94	0.97	0.50	0.50	0.50	0.50	**1**	**1**	**1**

Continued

	Accuracy										AUC									
	MLPL	MLPS	MLPA	WELS	WELSI	WELR	WELT	LSVL	LSVP	LSVR	MLPL	MLPS	MLPA	WELS	WELSI	WELR	WELT	LSVL	LSVP	LSVR
GNR	45.61	50.00	56.83	88.05	83.66	83.17	83.90	85.37	88.29	**93.66**	0.44	0.43	0.63	0.93	0.93	0.93	0.94	0.95	0.98	**1**
CORRS	92.44	83.90	87.07	89.27	82.68	84.63	83.41	84.15	87.07	**95.12**	0.97	0.90	0.94	0.90	0.92	0.90	0.92	0.95	0.99	**1**
CFS	**89.76**	83.66	87.32	86.34	82.44	83.17	84.15	83.41	86.59	87.56	0.95	0.90	0.92	0.88	0.91	0.91	0.91	0.92	0.94	**1**
OneR	89.27	86.34	88.78	89.02	80.49	85.61	84.39	83.41	87.32	**92.68**	0.95	0.91	0.93	0.90	0.91	0.92	0.92	0.93	0.98	**0.99**
GA	91.22	86.34	88.05	89.76	84.39	85.37	85.61	84.15	87.07	**94.63**	0.95	0.93	0.95	0.92	0.93	0.94	0.94	0.95	**0.99**	**0.99**
PCA	45.61	50.00	82.93	**88.05**	70.49	73.17	73.41	72.44	74.39	85.61	0.44	0.71	0.93	0.80	0.85	0.86	0.82	0.91	0.94	**0.99**
BLSMOTE-DATA																				
OD	84.88	83.17	85.85	50.00	50.00	50.00	50.00	96.34	93.41	**99.27**	0.94	0.93	0.93	0.50	0.50	0.50	0.50	0.99	0.98	**1**
SIGF	94.63	93.41	93.41	89.76	92.93	91.22	89.76	97.07	98.78	**99.27**	0.96	0.97	0.97	0.94	0.98	0.96	0.95	**1**	**1**	**1**
AG1	93.66	88.54	90.73	74.63	81.22	78.29	84.15	90.73	96.34	**96.59**	0.96	0.94	0.96	0.87	0.91	0.90	0.91	0.96	0.99	**1**
AG2	94.15	83.41	94.15	80.24	82.20	81.46	82.44	90.00	94.39	**98.54**	0.96	0.87	0.96	0.84	0.86	0.87	0.90	0.95	0.97	**1**
AG3	93.90	90.00	94.63	84.15	86.34	84.15	84.39	92.44	96.34	**98.78**	0.94	0.95	0.96	0.93	0.93	0.93	0.93	0.98	0.99	**1**
AG4	95.37	90.24	94.15	83.90	86.10	85.37	86.59	92.93	95.61	**98.78**	0.97	0.95	0.98	0.91	0.93	0.94	0.94	0.97	0.99	**1**
AG5	75.85	72.68	76.83	69.51	73.90	74.63	73.41	79.51	78.29	**88.54**	0.78	0.74	0.77	0.78	0.78	0.79	0.78	0.80	0.85	**0.95**
AG6	71.71	50.00	72.68	69.51	71.22	71.46	71.22	72.20	75.12	**93.66**	0.74	0.48	0.74	0.70	0.73	0.73	0.74	0.75	0.83	**0.98**
AG7	94.15	87.32	91.71	80.98	85.85	85.61	87.07	90.49	93.90	**98.78**	0.96	0.91	0.97	0.89	0.91	0.92	0.92	0.96	0.98	**1**
AG8	95.85	83.41	92.44	82.20	85.61	83.66	84.15	89.76	95.12	**99.27**	0.97	0.88	0.97	0.87	0.89	0.88	0.89	0.95	0.99	**1**
AG9	46.10	50.00	83.66	82.68	83.41	82.93	82.93	82.93	89.51	**99.02**	0.45	0.47	0.88	0.83	0.84	0.85	0.87	0.88	0.94	**1**
AG10	92.20	82.20	90.24	80.49	80.49	80.98	81.46	87.07	91.71	**99.02**	0.94	0.89	0.93	0.84	0.88	0.87	0.87	0.91	0.97	**1**
AG11	91.71	88.29	90.49	80.24	83.90	82.68	82.93	88.54	92.20	**99.27**	0.93	0.94	0.94	0.86	0.91	0.91	0.90	0.94	0.97	**1**
AG12	89.27	66.83	89.51	71.22	81.22	79.27	76.10	90.73	92.93	**95.37**	0.93	0.80	0.94	0.86	0.92	0.88	0.84	0.94	0.97	**0.99**
AG13	90.98	84.39	88.54	80.00	82.20	80.49	81.22	84.63	91.46	**98.78**	0.94	0.90	0.94	0.87	0.89	0.88	0.87	0.92	0.97	**1**
INFG	44.15	50.00	50.00	50.00	50.00	50.00	50.00	96.83	96.10	**99.27**	0.42	0.43	0.43	0.50	0.50	0.50	0.50	0.99	0.99	**1**
GNR	47.07	50.00	73.41	88.78	87.56	88.54	90.73	91.95	97.32	**99.02**	0.46	0.45	0.86	0.94	0.95	0.95	0.95	0.95	**1**	**1**
CORRS	93.66	85.37	89.51	81.46	84.88	84.88	84.88	87.32	92.20	**98.29**	0.96	0.91	0.95	0.92	0.91	0.91	0.92	0.97	0.98	**1**
CFS	89.51	85.85	85.61	79.76	80.24	80.00	80.73	87.32	92.20	**97.32**	0.95	0.90	0.92	0.84	0.88	0.86	0.88	0.93	0.99	**1**
OneR	91.95	85.61	90.98	84.63	85.37	84.63	84.63	87.32	94.88	**97.32**	0.95	0.92	0.95	0.90	0.94	0.92	0.92	0.95	0.98	**1**
GA	93.41	85.37	93.41	85.61	86.10	85.12	85.37	87.56	96.59	**98.78**	0.95	0.93	0.96	0.92	0.93	0.92	0.93	0.97	0.99	**1**
PCA	46.10	50.00	85.85	75.61	76.34	75.37	78.05	81.22	89.76	**95.85**	0.45	0.80	0.94	0.89	0.89	0.84	0.90	0.93	0.96	**0.99**

Table 12. Accuracy and AUC for Anti-pattern 1: Advance Level of Classifiers . Best performance value in bold.

This test is used to test our considered null hypothesis "There is no significant impact on the performance of anti-patter models after applying feature selection techniques". The considered hypothesis is only accepted if the calculated p-value using WSRT is less than 0.05. Figure 5 shows the result of WSRT on different pairs of feature sets, i.e., × symbol indicates that the p-value$\leq$0.05, and □ symbol indicates that the p-value>0.05. According to Fig. 5, the predictive ability of the models is significantly impacted by using different sets of features. After finding the impact of feature selection techniques, we have also applied Friedman's mean rank (FMR) to find the best sets of features for anti-pattern prediction. The last column of Table 14 shows the FMR for the aggregation measures and the various applied feature selection techniques. According to FRM, the SIGF has the lowest mean rank of 5.97. Hence, we conclude that the models trained by taking selected sets of features using SIGF have a significantly better ability of prediction as compared to other techniques. Similarly, PCA has the highest mean AUC rank, 17.90, indicating that the model developed with features selected by PCA will have the worst performance.

RQ 2:	What is the significant impact of considering reduced sets of features as input on the performance of models?
ANS:	The experimental findings based on Figs. 4a, b, 5 and Table 14 confirmed that the models trained by taking selected sets of features can predict significantly better than all features.

Sampling techniques

In this experiment, we have also considered five types of data imbalance techniques such as SMOTE, BLSMOTE, SVMSOMTE, ADASYN, and UPSAM to tackle the class imbalance problem, and the resulting balanced datasets are used as training data for anti-pattern prediction models. The significance and reliability of these employed sampling strategies were determined using statistical and AUC analyses.

Comparison of sampling techniques using descriptive statistics and box-plots: Figure 6 shows the box-plot diagram for accuracy and the AUC of the models trained on sampled datasets. These sample datasets are generated using five different sampling techniques. The descriptive statistics for AUC and accuracy for sampling techniques considered are summarized in Table 15. According to Fig. 6 and Table 15, the models trained on sampled data using upsampling (UPSAM) with a mean AUC of 0.87 achieved better results. In contrast, the model developed with the original data with a mean AUC of 0.70 has the worst performance. ADASYN and SMOTE showed the worst performance among the data sampling techniques applied, with mean AUC values of 0.83 and 0.83, respectively.

Comparison of different sampling technique: Wilcoxon Signed Rank Test (WSRT) with Friedman mean rank (FMR):

In this experiment, we have also employed two statistical tests for hypothesis analysis: Wilcoxon Signed Rank Test and Friedman Test. Initially, we applied WSRT to verify the impact of sampling techniques on the performance of anti-pattern prediction models. This test is used to test our considered null hypothesis "There is no significant impact on the performance of anti-patter models after training on balanced data". Figure 7 shows the result of WSRT on different pairs of sampling techniques, i.e., × symbol indicates that the p-value$\leq$0.05, and □ symbol indicates that the p-value>0.05. The information present in Fig. 7 suggested that the null hypothesis was rejected for all comparable sampling technique pairs. Hence, we concluded that the predictive ability of the models is significantly impacted by using sampling techniques. After verifying the conclusion like "the performance of the models significantly improves after training on sampled data", we have used the Friedman test to find the best sampling techniques. The lower rank of the Friedman test represents the best results. Table 15 shows the Friedman test results for various data sampling techniques. From Table 15, we infer that UPSAM has the best performance with a mean AUC rank of 2.13, whereas the model developed with the original dataset has the worst performance with a mean rank of 5.41.

RQ 3:	What is the significant impact of sampling techniques on the predictability of anti-pattern prediction models?
ANS:	The experimental findings based on Figs. 6, 7 and Table 15 confirmed that the predictive ability of the models is significantly impacted by using sampling techniques. The performance of the models significantly improves after training on sampled data.

Classification techniques

In this work, 33 classifiers varying from general machine learning classifiers to advance deep learning classifiers have been employed to train models for detecting web service anti-patterns. We computed the implications and dependabilities of these classifiers using box plots, descriptive statistics, and statistical test analyses on different anti-patterns.

Comparison of different classification techniques using descriptive statistics and box plots: Figure 8 shows the AUC and accuracy box plots for the different categories of classifier techniques. According to Fig. 8, we can conclude the following:

- Among the general classifiers category, KNN shows the best performance with a mean AUC value of 0.92. In contrast, the Support Vector Machine with the linear kernel (SVC-LIN) offers the worst performance, with a mean AUC value of 0.62.

	Accuracy									AUC								
	BAG	RF	EXTR	AdaB	GraB	DL1	DL2	DL3	DL4	BAG	RF	EXTR	AdaB	GraB	DL1	DL2	DL3	DL4
ORG-DATA																		
OD	90.71	91.15	92.48	90.27	91.59	**92.92**	92.04	92.04	**92.92**	0.81	0.82	0.84	0.85	0.67	**0.88**	0.87	0.86	**0.88**
SIGF	89.82	89.82	91.59	91.59	90.71	91.59	**92.04**	**92.04**	**92.04**	0.82	0.80	0.80	0.78	0.66	**0.88**	0.87	0.86	0.87
AG1	91.59	92.48	91.59	**92.92**	92.04	90.71	90.71	90.71	90.71	0.85	0.84	0.73	0.82	0.62	**0.89**	0.87	0.86	0.86
AG2	**90.71**	89.38	89.82	88.94	89.38	**90.71**	**90.71**	**90.71**	**90.71**	0.76	0.75	0.75	0.76	**0.8**	0.69	0.75	0.76	0.76
AG3	90.71	90.71	**91.15**	89.82	**91.15**	90.71	90.71	90.71	90.71	0.77	0.82	0.81	0.82	0.68	0.81	**0.84**	0.83	0.83
AG4	90.27	89.82	**91.59**	89.82	91.15	89.82	90.71	90.71	90.71	0.77	**0.85**	0.81	0.82	0.68	0.83	0.84	0.83	0.83
AG5	**90.71**	88.05	89.38	89.82	**90.71**	**90.71**	**90.71**	**90.71**	**90.71**	0.70	**0.71**	0.68	0.70	0.64	0.41	0.44	0.42	0.40
AG6	**90.71**	88.94	86.73	89.38	89.38	**90.71**	**90.71**	**90.71**	**90.71**	**0.74**	0.67	0.55	0.73	0.69	0.44	0.50	0.63	0.40
AG7	91.59	91.59	92.04	90.27	89.82	**92.92**	89.38	90.27	90.71	0.82	0.83	0.76	0.85	0.59	**0.87**	0.84	0.84	0.84
AG8	90.71	90.27	91.59	91.59	91.59	**92.04**	90.27	90.71	90.71	0.86	0.82	0.81	0.82	0.60	0.83	0.86	**0.87**	0.86
AG9	90.27	89.82	**91.59**	**91.59**	89.38	90.71	90.71	90.71	90.71	0.80	0.71	0.77	**0.85**	0.83	0.68	0.60	0.61	0.59
AG10	90.71	**91.15**	90.71	**91.15**	87.61	90.71	90.71	90.71	90.71	0.81	0.75	0.74	**0.83**	0.75	0.71	0.75	0.73	0.73
AG11	88.50	**91.59**	90.71	90.27	90.27	90.71	90.71	90.71	90.71	**0.83**	0.73	0.79	0.82	**0.83**	0.75	0.81	0.80	0.79
AG12	**91.59**	90.71	90.71	90.71	88.94	90.71	90.71	90.71	90.71	0.86	0.77	0.79	0.84	0.61	0.69	**0.87**	0.75	0.77
AG13	89.82	91.59	**92.48**	89.82	90.27	90.71	90.71	90.71	90.71	**0.84**	0.79	0.74	0.82	0.83	0.75	0.83	0.81	0.80
INFG	91.15	92.48	91.59	**92.92**	92.04	91.59	91.59	91.59	91.15	0.82	0.85	0.84	0.85	0.73	**0.88**	**0.88**	0.87	0.86
GNR	91.59	92.04	91.15	**92.92**	89.38	90.71	90.71	90.71	90.71	**0.91**	0.88	0.83	0.87	0.52	0.85	0.80	0.89	0.88
CORR	90.71	92.48	**93.81**	92.92	88.94	91.15	90.71	90.71	90.71	**0.88**	0.80	0.85	0.87	0.64	0.86	0.84	0.84	0.82
CFS	90.71	92.48	**92.92**	92.48	92.04	91.15	91.15	90.71	90.71	0.92	0.86	0.83	0.84	0.71	**0.94**	0.91	0.89	0.88
OneR	92.04	91.59	92.04	**92.92**	**92.92**	91.15	90.71	90.71	90.71	**0.9**	0.85	0.83	0.79	0.64	0.79	0.84	0.86	0.83
GA	**92.04**	**92.04**	91.15	91.15	91.59	91.15	90.71	90.27	90.71	**0.91**	0.86	0.84	0.87	0.65	0.87	0.84	0.87	0.81
PCA	**90.71**	90.27	90.27	90.27	88.94	**90.71**	**90.71**	**90.71**	**90.71**	**0.82**	0.81	0.78	0.75	0.75	0.47	0.53	0.52	0.44
SMOTE-DATA																		
OD	90.00	95.12	**96.1**	91.71	91.46	93.17	95.12	94.15	94.63	0.97	**0.98**	**0.98**	0.97	0.97	0.97	**0.98**	**0.98**	**0.98**
SIGF	89.76	94.15	**96.34**	92.93	93.90	94.39	94.88	95.61	95.12	0.96	0.98	**0.99**	0.97	0.97	0.97	0.97	0.97	0.98
AG1	89.27	90.49	**92.93**	90.00	89.02	85.85	86.59	87.56	86.59	0.96	0.96	**0.97**	0.94	0.96	0.92	0.90	0.93	0.93
AG2	86.34	92.68	**93.41**	87.80	88.05	76.83	81.71	81.46	82.68	0.93	**0.97**	**0.97**	0.94	0.95	0.83	0.87	0.86	0.86
AG3	89.27	92.93	**96.1**	88.54	90.73	83.90	86.34	88.54	87.56	0.96	**0.98**	**0.98**	0.95	0.95	0.90	0.92	0.92	0.92
AG4	89.27	**95.61**	95.12	91.71	90.98	82.20	85.12	88.78	89.51	0.95	0.97	**0.98**	0.96	0.95	0.90	0.92	0.94	0.94
AG5	80.00	84.88	**85.37**	74.15	78.05	72.44	70.00	67.32	54.63	0.88	**0.92**	0.91	0.83	0.84	0.77	0.74	0.71	0.53
AG6	81.95	85.61	**87.32**	80.24	80.98	70.49	54.39	48.29	51.22	0.87	0.90	**0.92**	0.88	0.87	0.70	0.56	0.49	0.57
AG7	91.71	92.20	**94.63**	89.76	91.22	86.34	89.51	89.76	89.51	**0.97**	**0.97**	**0.97**	0.95	0.94	0.93	0.94	0.94	0.94
AG8	90.49	93.90	**95.61**	87.56	87.80	87.80	90.24	90.49	91.22	0.96	**0.98**	**0.98**	0.96	0.95	0.93	0.94	0.95	0.94
AG9	85.37	93.66	**95.37**	90.73	90.00	77.80	78.05	77.80	64.63	0.95	**0.98**	**0.98**	0.96	0.96	0.82	0.81	0.81	0.73
AG10	84.88	88.78	**92.93**	83.66	83.90	79.27	81.71	81.46	74.63	0.92	0.93	**0.96**	0.91	0.89	0.87	0.87	0.85	0.82
AG11	87.07	89.02	**93.41**	85.85	84.39	82.93	84.88	86.10	85.85	0.94	**0.97**	**0.97**	0.93	0.93	0.90	0.90	0.90	0.89
AG12	87.07	91.71	**93.66**	88.05	87.80	81.22	83.17	84.39	85.61	0.95	**0.97**	**0.97**	0.93	0.93	0.91	0.91	0.90	0.89
AG13	84.88	91.71	**93.17**	86.34	83.90	82.44	84.15	85.61	85.37	0.94	0.96	**0.98**	0.92	0.91	0.90	0.90	0.90	0.90
INFG	88.29	91.71	94.15	91.46	90.73	90.00	93.17	93.17	**94.39**	0.96	**0.97**	**0.97**	**0.97**	0.96	0.96	**0.97**	**0.97**	0.96
GNR	88.05	88.05	**90.49**	85.85	86.10	86.83	87.56	88.78	81.22	0.95	**0.96**	**0.96**	0.93	0.94	0.93	0.92	0.93	0.86
CORR	89.27	91.22	**91.46**	88.29	88.05	85.85	85.85	86.59	86.10	0.95	**0.96**	**0.96**	0.94	0.93	0.91	0.90	0.88	0.89
CFS	86.34	89.51	**91.46**	89.76	89.02	83.90	85.12	86.10	78.54	0.93	0.95	**0.96**	0.93	0.94	0.91	0.90	0.91	0.88
OneR	89.02	89.51	**92.44**	90.00	90.00	86.83	87.07	78.54	78.78	**0.96**	**0.96**	**0.96**	0.95	0.95	0.92	0.93	0.89	0.89
GA	89.76	**92.44**	91.46	90.49	89.27	86.83	86.59	86.34	79.76	0.95	**0.96**	**0.96**	0.94	0.94	0.92	0.93	0.92	0.90
PCA	88.05	**91.95**	91.71	85.85	86.83	75.85	74.39	64.88	56.10	0.94	**0.97**	**0.97**	0.92	0.93	0.81	0.88	0.76	0.64
SMOTE-DATA																		
BLSMOTE-DATA																		
OD	89.27	94.63	**96.34**	93.17	91.46	92.93	93.90	93.66	95.12	0.97	**0.98**	**0.98**	0.96	0.96	0.97	0.97	0.97	0.97
SIGF	89.51	92.93	**95.85**	91.22	90.00	92.68	92.93	93.17	93.66	0.96	**0.98**	**0.98**	0.97	0.96	0.97	**0.98**	0.97	**0.98**
AG1	91.71	93.41	**94.88**	91.22	92.93	88.78	89.02	91.22	90.24	0.97	0.97	**0.98**	0.97	0.97	0.94	0.94	0.94	0.94
AG2	90.24	95.37	**96.34**	93.41	94.63	85.85	88.78	90.00	92.44	**0.97**	**0.97**	**0.97**	**0.97**	0.96	0.90	0.91	0.93	0.94
AG3	91.22	94.63	**96.1**	92.20	92.68	87.56	89.76	90.49	91.22	0.97	0.97	**0.98**	0.96	0.97	0.94	0.95	0.95	0.95
AG4	91.95	93.41	**96.1**	93.17	93.66	88.54	90.49	90.73	91.71	0.97	**0.98**	**0.98**	0.97	0.97	0.94	0.95	0.95	0.95
AG5	79.27	**85.37**	84.63	73.90	80.73	73.41	69.27	70.24	48.54	0.90	**0.91**	0.90	0.82	0.85	0.74	0.73	0.72	0.49

Continued

	Accuracy									AUC								
	BAG	RF	EXTR	AdaB	GraB	DL1	DL2	DL3	DL4	BAG	RF	EXTR	AdaB	GraB	DL1	DL2	DL3	DL4
AG6	82.93	86.10	**87.8**	80.98	78.05	72.68	60.49	63.17	54.63	0.88	0.92	**0.93**	0.90	0.87	0.76	0.67	0.69	0.55
AG7	89.76	91.71	**94.63**	88.54	88.29	88.05	89.27	89.27	90.49	0.96	0.96	**0.97**	0.94	0.93	0.92	0.93	0.93	0.93
AG8	89.02	93.90	**95.12**	88.05	88.29	85.37	87.32	87.56	87.32	0.96	0.96	**0.98**	0.94	0.94	0.89	0.90	0.90	0.91
AG9	87.32	**95.85**	95.37	91.22	92.44	83.66	83.90	76.34	76.59	0.96	**0.98**	**0.98**	0.97	0.97	0.85	0.85	0.83	0.83
AG10	87.07	91.46	**92.93**	88.29	85.12	81.46	84.63	85.12	78.05	0.94	**0.96**	**0.96**	0.94	0.89	0.88	0.88	0.88	0.86
AG11	88.29	92.93	**94.39**	86.10	86.59	84.88	87.07	88.05	87.80	0.95	0.97	**0.98**	0.94	0.94	0.92	0.92	0.93	0.92
AG12	89.76	**92.68**	**92.68**	89.76	90.00	83.17	86.83	87.56	87.32	0.95	0.96	**0.98**	0.95	0.93	0.92	0.92	0.92	0.92
AG13	87.32	90.73	**93.66**	85.61	86.83	81.71	83.66	84.39	77.07	0.93	0.96	**0.97**	0.91	0.92	0.89	0.90	0.90	0.86
INFG	89.51	94.39	**95.61**	93.41	92.20	92.20	92.20	92.93	92.68	0.97	**0.98**	**0.98**	0.97	0.96	0.96	0.97	**0.98**	0.97
GNR	91.71	**94.63**	**94.63**	92.20	92.68	91.95	91.95	91.71	90.73	0.97	**0.98**	**0.98**	0.96	0.96	0.95	0.95	0.94	0.93
CORR	89.02	92.93	**93.66**	92.68	91.95	86.34	86.34	86.83	85.37	0.96	**0.97**	**0.97**	0.96	0.95	0.92	0.93	0.92	0.91
CFS	87.07	90.00	**92.2**	87.07	88.78	82.44	84.15	84.63	75.61	0.93	0.94	**0.95**	0.93	0.92	0.89	0.89	0.89	0.86
OneR	91.95	92.20	**92.68**	88.05	91.71	86.10	86.10	90.00	77.32	**0.97**	0.96	**0.97**	0.95	0.96	0.93	0.94	0.95	0.87
GA	93.17	93.66	**94.15**	92.20	92.68	87.07	87.56	89.76	90.24	0.97	**0.98**	**0.97**	0.96	0.96	0.94	0.94	0.95	0.94
PCA	88.05	91.71	**94.39**	87.80	87.07	79.51	80.98	59.02	51.22	0.94	**0.97**	**0.97**	0.93	0.94	0.88	0.90	0.69	0.52

Table 13. Accuracy and AUC for Anti-pattern 1: ensemble Classifiers and Deep-Learning. Best performance value in bold.

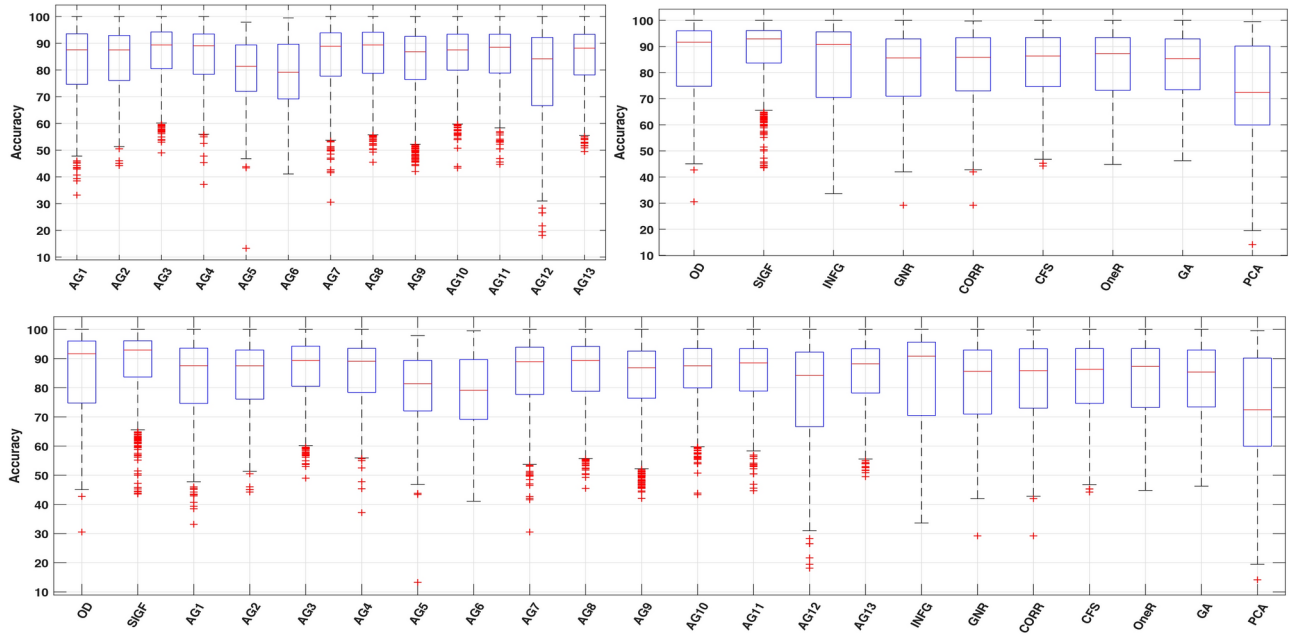
- Among the ensemble classifiers employed for training the models for detection of anti-patterns, the extra tree classifier (EXTR) and Random Forest (RF) classifiers are showing the best performance with a mean AUC value of 0.94, and the Gradient Boosting classifier (GraB) is delivering the worst performance with a mean AUC of 0.88.
- Of all the deep learning algorithms with varying hidden layers, DL2 shows the best performance with a mean AUC value of 0.84, and DL4 offers the worst performance with a mean AUC value of 0.81. DL3 performance is similar to that of the model developed using DL2.
- Over the advanced ML classifiers used for developing models for detecting web service anti-patterns, LSSVM-RBF shows the best performance with a mean AUC value of 0.99. In contrast, the models trained using ELM-LIN show the worst performance, with a mean AUC value of 0.70.Figure 9 shows the AUC and accuracy values for all the classifier techniques combinedly. The descriptive statistics for all the classifier techniques are depicted in Table 16. From Figs. 9 and Table 16, we infer that the models trained using LSSVM-RBF are showing the best performance with a mean AUC value of 0.99. LSSVM-Poly, RF, and EXTR perform better after LSSVM-RBF with a mean AUC value of 0.95, 0.94, and 0.94, respectively. SVC-LIN is delivering the worst performance with a mean AUC value of 0.63.

Comparison of different classification techniques: Wilcoxon Signed Rank Test (WSRT) with Friedman mean rank (FMR): Similar to the previous subsections, we also have the Wilcoxon Test and the Friedman test to compute statistically significant differences among various pairs of classifier techniques. Initially, we applied WSRT to verify the impact of different classifiers on the performance of anti-pattern prediction models. This test is used to test our considered null hypothesis "There is no significant impact on the performance of anti-patter models after changing classifiers". Figure 10 shows the result of WSRT on different pairs of sampling techniques, i.e., $\times$ symbol indicates that the p-value$\leq$0.05, and $\square$ symbol indicates that the p-value>0.05. According to Fig. 10, the predictive ability of the models trained using different classifiers is not significantly the same. Table 16 shows the Friedman test results for various classifier techniques considered in this work. From Table 16, we infer that LSSVM-RBF has the best performance with a mean rank of 1.18, whereas the SVC-LIN classifier technique has the worst performance with a mean rank of 27.60.

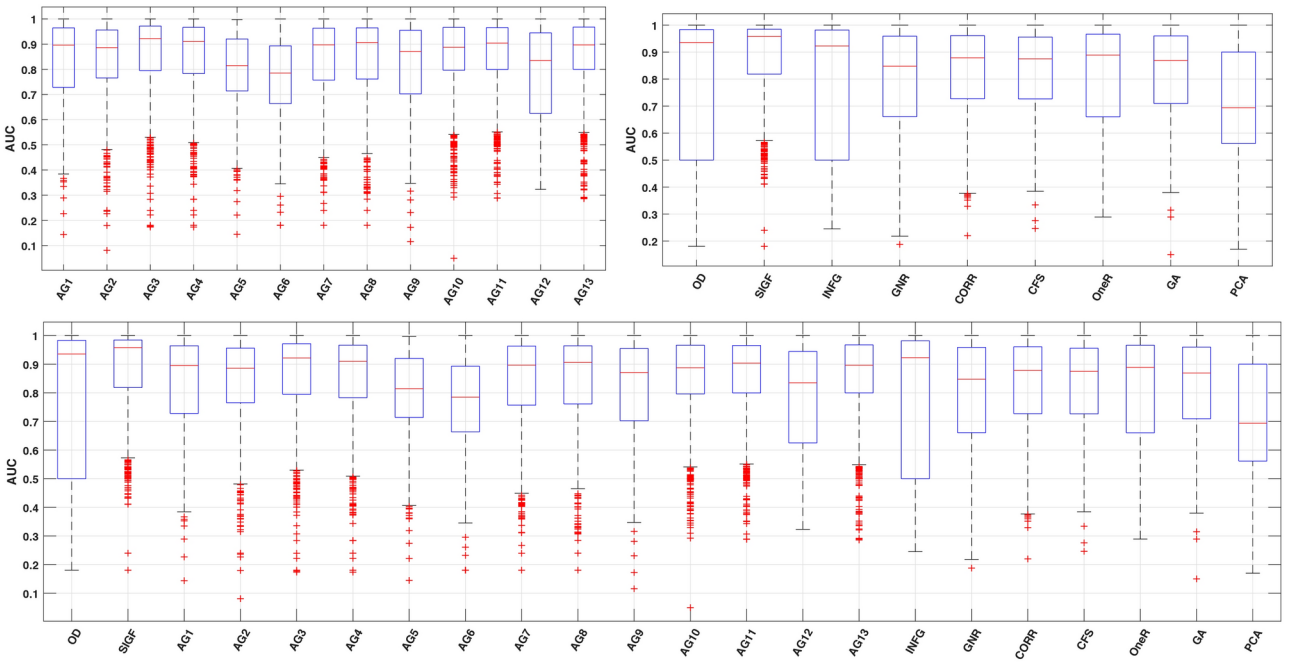
RQ 4:	What effect do different classifiers have on predicting anti-patterns using source code metrics?
ANS:	The experimental findings based on Figs. 9, 10, and Table 16 confirmed that the predictive ability of the models trained using different classification techniques is significantly different. The performance of the models significantly improves after changing the classification techniques.

Discussion of results

In this work, extensive experimentation by using different variants of aggregation measures, feature selections, data sampling, and classifiers has been made, and a solution for developing such models to predict the anti-pattern using object-oriented metrics with improved performance and predictability power is proposed. In general, it was observed that the prediction models with 0.7 AUC value have the ability to predict class on unseen



(a) Accuracy



(b) AUC

Fig. 4. Accuracy and AUC boxplots of feature selection techniques.

patterns i.e., the models with 0.7 AUC are acceptably by the community. The experimental results obtained using the proposed framework confirm that the trained models delivered a greater than 0.7 AUC and have the ability to predict anti-patterns on an unseen WSDL file. We have already presented the AUC values of the models trained for anti-pattern 1 using sets of features with different classifiers on both original and balanced data. The highest possible AUC value for all classifiers with the application of different combinations of sampling and feature selection techniques attained greater than 0.9; this proves the greater predictability of developed anti-pattern prediction models. The classifier post-application of feature selection and sampling techniques has outperformed with an AUC value of 1.

	Accuracy						AUC						Friedman
	Mean	Min	Median	Max	Q3	Q1	Mean	Min	Median	Max	Q3	Q1	Rank
OD	83.35	30.53	91.64	100.00	96.01	74.75	0.80	0.18	0.94	1.00	0.98	0.50	8.48
SIGF	88.40	43.56	92.92	100.00	96.10	83.66	0.88	0.18	0.96	1.00	0.98	0.82	5.97
AG1	82.73	33.19	87.56	100.00	93.56	74.63	0.83	0.14	0.90	1.00	0.96	0.73	12.27
AG2	83.86	44.25	87.53	100.00	92.92	76.10	0.84	0.08	0.89	1.00	0.96	0.77	12.40
AG3	86.20	49.00	89.38	100.00	94.25	80.49	0.86	0.17	0.92	1.00	0.97	0.79	8.88
AG4	85.36	37.17	89.08	100.00	93.49	78.40	0.85	0.17	0.91	1.00	0.97	0.78	10.31
AG5	80.17	13.27	81.39	97.89	89.39	72.02	0.80	0.14	0.81	1.00	0.92	0.71	14.30
AG6	78.25	41.09	79.14	99.51	89.65	69.18	0.76	0.18	0.79	1.00	0.89	0.66	15.16
AG7	84.56	30.53	88.93	100.00	93.90	77.70	0.84	0.18	0.90	1.00	0.96	0.76	10.53
AG8	85.33	45.50	89.38	100.00	94.15	78.78	0.84	0.18	0.91	1.00	0.96	0.76	9.74
AG9	82.49	42.04	86.84	100.00	92.57	76.42	0.81	0.12	0.87	1.00	0.95	0.70	12.51
AG10	85.43	43.32	87.53	100.00	93.41	79.94	0.85	0.05	0.89	1.00	0.97	0.80	9.77
AG11	85.34	44.69	88.50	100.00	93.40	78.87	0.85	0.29	0.90	1.00	0.97	0.80	9.77
AG12	79.29	18.14	84.20	100.00	92.20	66.67	0.78	0.32	0.84	1.00	0.94	0.62	15.06
AG13	85.09	49.51	88.21	100.00	93.36	78.17	0.85	0.29	0.90	1.00	0.97	0.80	9.97
INFG	82.48	33.63	90.83	100.00	95.61	70.49	0.80	0.25	0.92	1.00	0.98	0.50	9.35
GNR	81.24	29.20	85.61	100.00	92.92	70.98	0.80	0.19	0.85	1.00	0.96	0.66	12.75
CORR	82.37	29.20	85.85	99.76	93.36	73.00	0.82	0.22	0.88	1.00	0.96	0.73	12.38
CFS	83.18	44.24	86.34	100.00	93.41	74.65	0.83	0.25	0.88	1.00	0.96	0.73	12.06
OneR	82.10	44.80	87.30	100.00	93.43	73.21	0.81	0.29	0.89	1.00	0.97	0.66	11.50
GA	82.49	46.24	85.37	100.00	92.92	73.41	0.82	0.15	0.87	1.00	0.96	0.71	11.95
PCA	72.86	14.16	72.44	99.53	90.18	59.95	0.71	0.17	0.69	1.00	0.90	0.56	17.90

Table 14. Employed feature selection techniques' descriptive statistics.

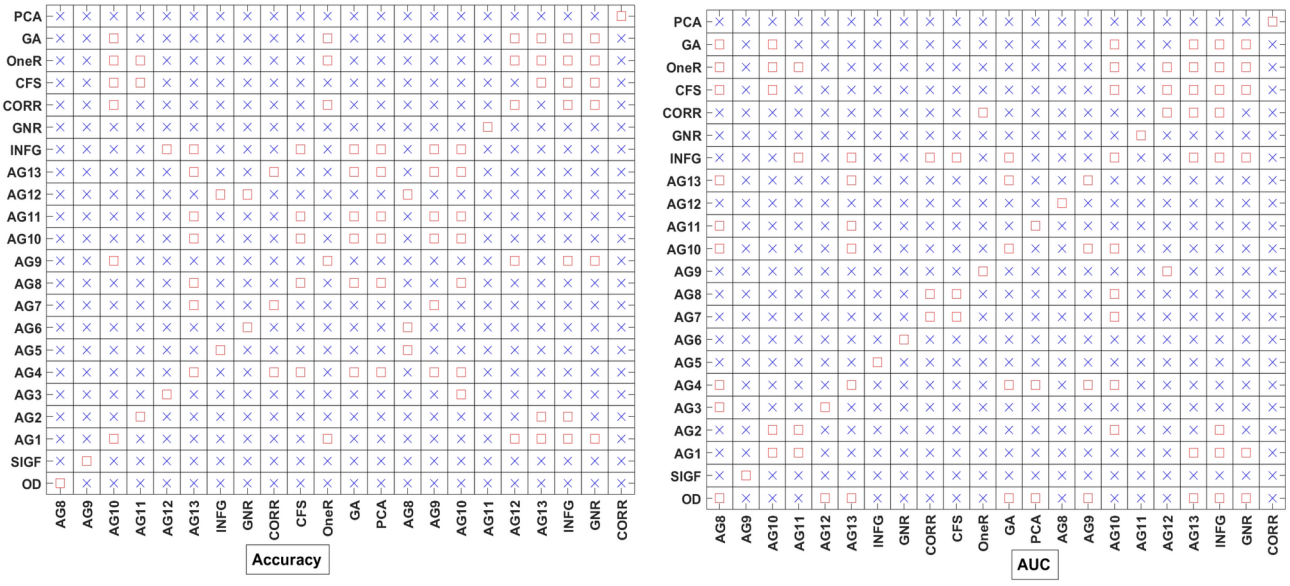


Fig. 5. Statistical test results of feature selection techniques: accuracy and AUC.

Conclusion

The developed web service anti-pattern prediction models using object-oriented metrics help in building quality web-based applications by identifying anti-patterns at the initial stage of Software Development. This research represents a significant step forward in the development of effective anti-pattern prediction models by dealing with feature selection, aggregation measures, and the class imbalance problem efficiently. The developed anti-pattern prediction models use different variants of classifiers, and their performance has been measured against five different variants of anti-patterns. The proposed framework was validated using 226 WSDL files collected from various domains such as finance, tourism, health, education, etc. The focused insights of this research are:

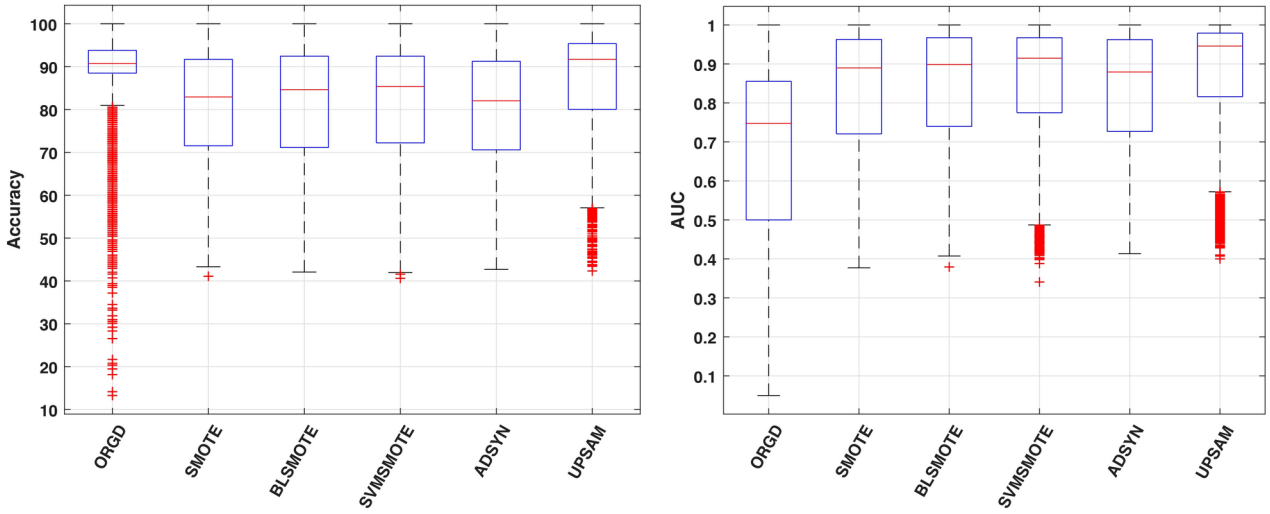


Fig. 6. Accuracy and AUC boxplots of sampling techniques.

	Accuracy						AUC						Friedman
	Mean	Min	Median	Max	Q3	Q1	Mean	Min	Median	Max	Q3	Q1	Rank
ORGD	87.58	13.27	90.71	100.00	93.81	88.50	0.70	0.05	0.75	1.00	0.86	0.50	5.41
SMOTE	80.33	41.09	82.93	100.00	91.71	71.53	0.83	0.38	0.89	1.00	0.96	0.72	3.41
BLSMOTE	81.14	42.08	84.63	100.00	92.44	71.13	0.84	0.38	0.90	1.00	0.97	0.74	3.08
SVMSMOTE	82.07	40.60	85.34	100.00	92.44	72.22	0.85	0.34	0.91	1.00	0.97	0.77	3.23
ADSYN	79.80	42.68	82.03	100.00	91.25	70.59	0.83	0.41	0.88	1.00	0.96	0.73	3.74
UPSAM	86.14	42.33	91.71	100.00	95.37	80.05	0.87	0.40	0.95	1.00	0.98	0.82	2.13

Table 15. All Sampling Techniques' Descriptive Statistics.

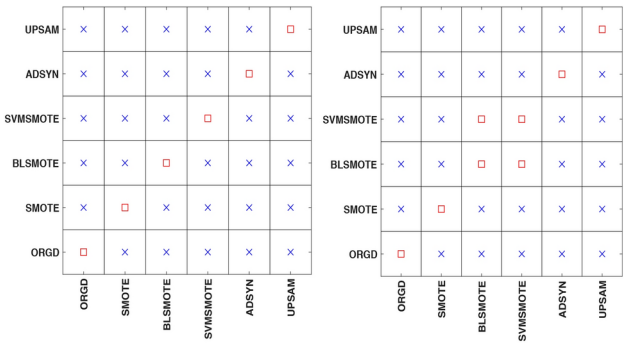
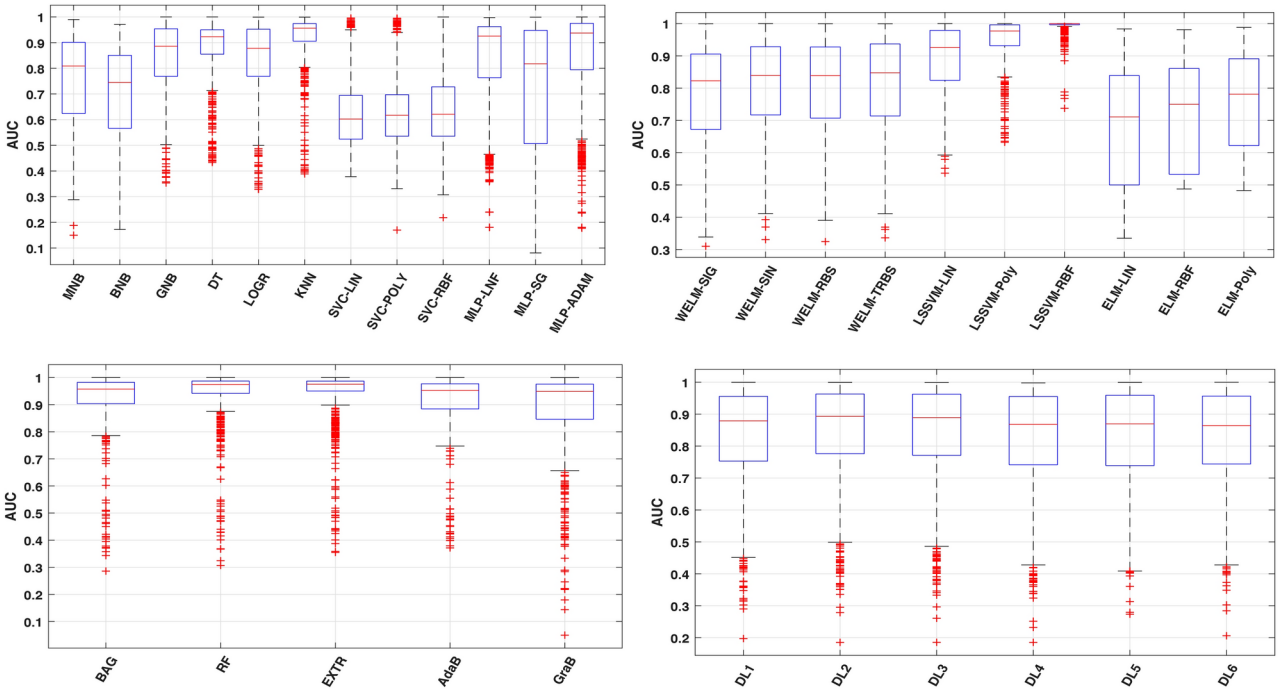
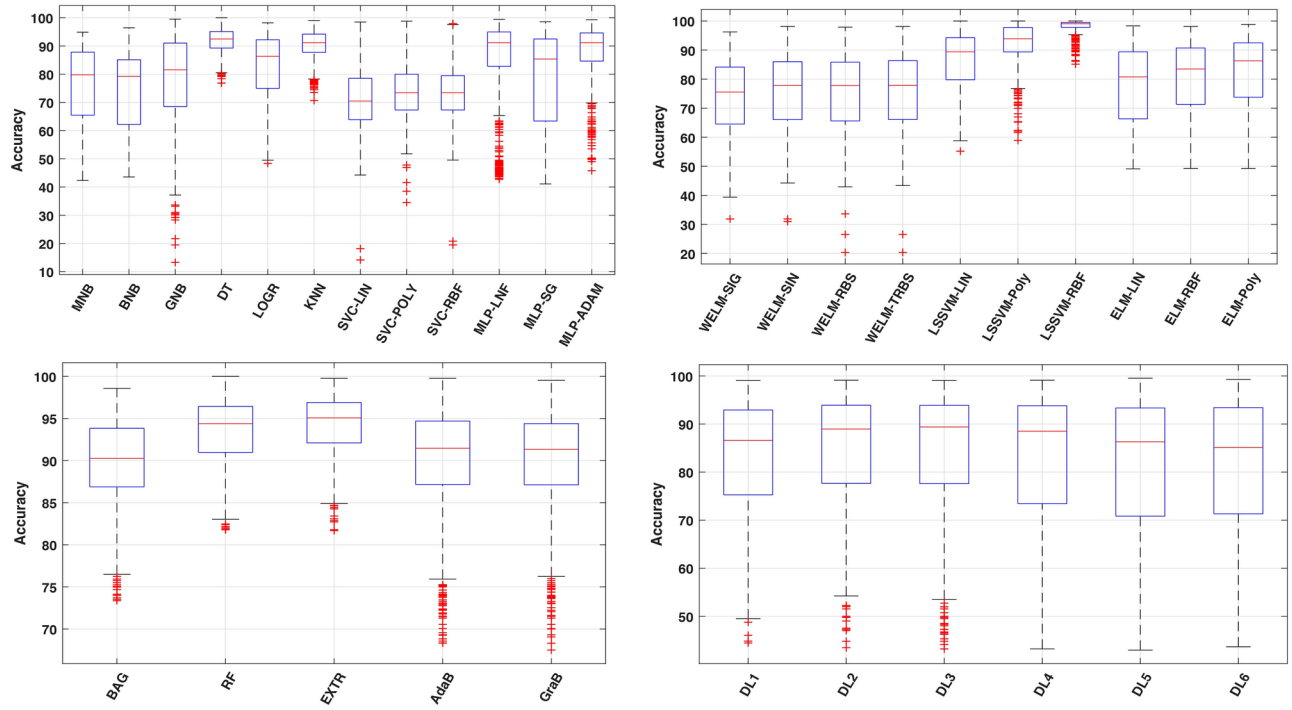


Fig. 7. Statistical test results: Accuracy and AUC: Sampling Techniques.

- For most anti-patterns, adopting sampling methods such as SMOTE, BLSMOTE, SVMSMOTE, and USAM improves the predictability of developed models.
- Employing the different variants of aggregation measures with feature selection strategies over balanced datasets reduces computational effort and improves the overall performance of anti-pattern prediction models.
- In comparison to other employed sampling techniques to handle the data imbalance, the UPSAM technique outperformed by gaining the highest mean Accuracy & AUC of 86.14% & 0.87, respectively
- Experimental results suggested that the models trained by selected sets of features using SIGF performance best compared to other employed feature selection techniques by attaining 88.40% Accuracy & 0.88 AUC. This finding confirmed that there exist irrelevant features.
- The LSSVM with RBF kernel classifier stands first among all other classifiers.
- Post implication of feature selection techniques study indicates a reduction of 97% irrelevant features from the original dataset while pertaining improved performance of anti-pattern models trained against all metrics.

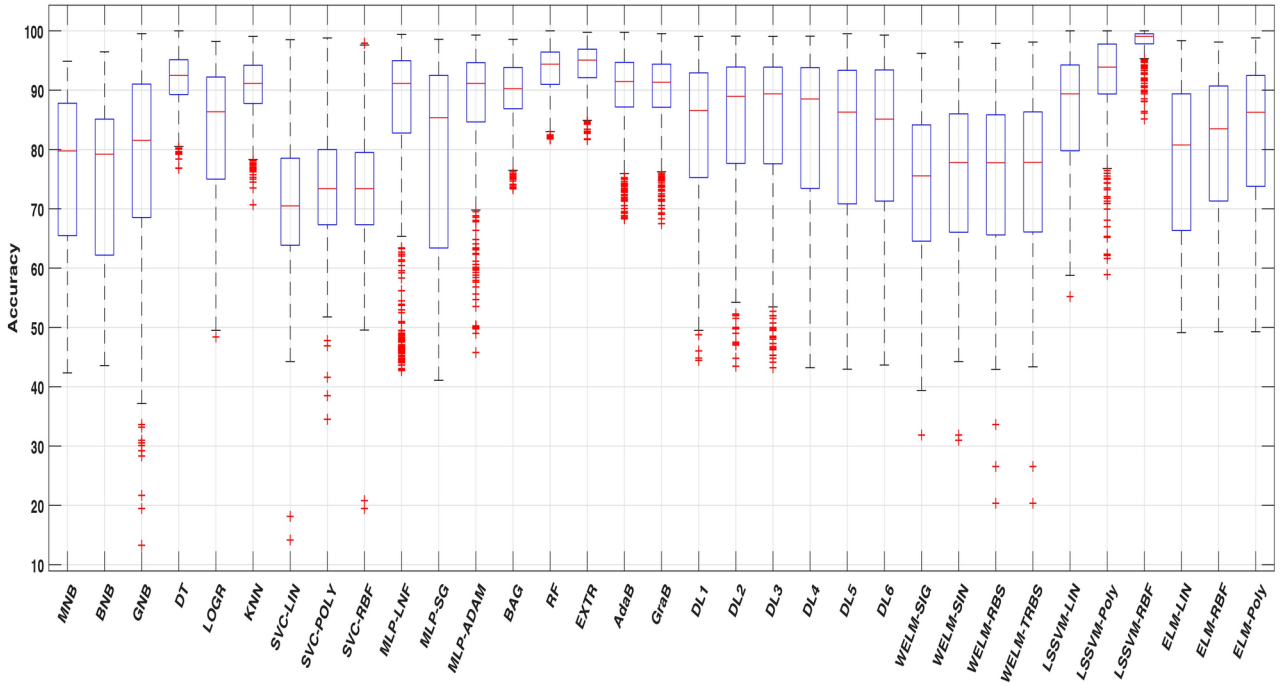


(a) Accuracy

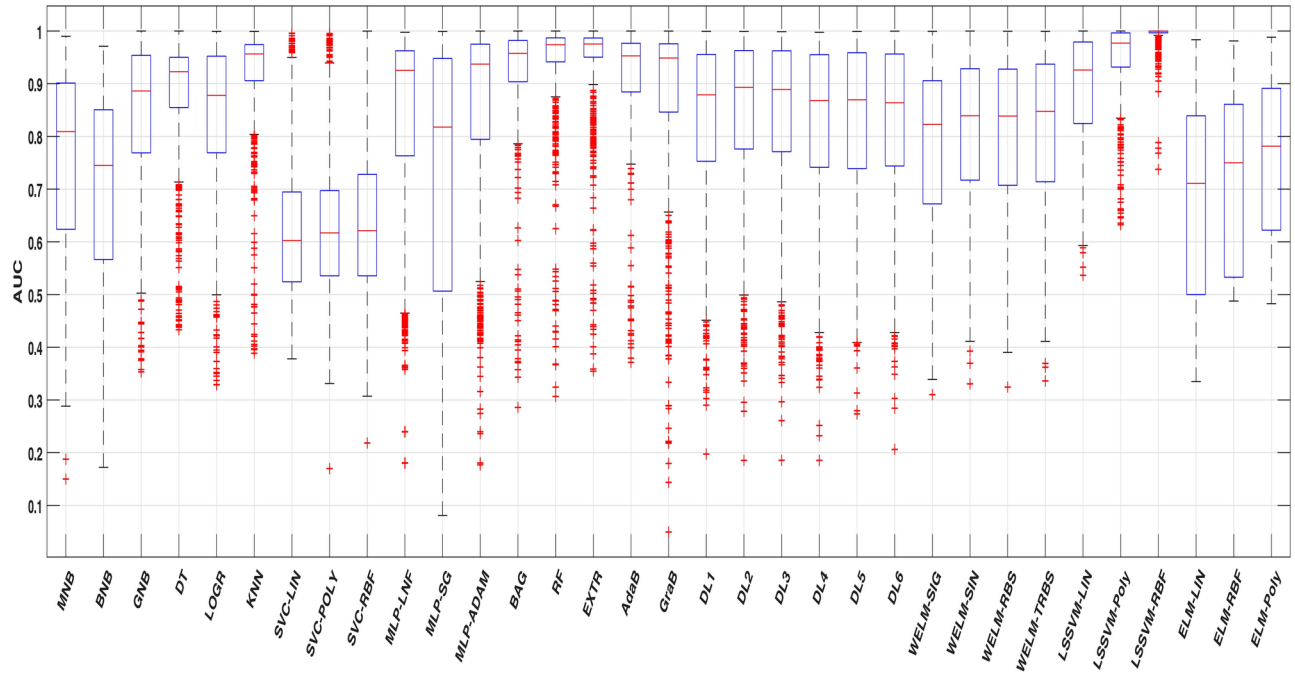


(b) AUC

Fig. 8. Accuracy and AUC Boxplots of different classifiers.



(a) Accuracy



(b) AUC

Fig. 9. Full comparison accuracy and AUC boxplots of different classifiers.

	Accuracy						AUC						Friedman
	Mean	Min	Median	Max	Q3	Q1	Mean	Min	Median	Max	Q3	Q1	Rank
MNB	76.42	42.33	79.76	94.88	87.80	65.48	0.76	0.15	0.81	0.99	0.90	0.62	24.71
BNB	74.62	43.56	79.20	96.46	85.12	62.20	0.70	0.17	0.74	0.97	0.85	0.57	27.34
GNB	78.29	13.27	81.55	99.53	91.03	68.54	0.84	0.35	0.89	1.00	0.95	0.77	16.19
DT	91.78	76.85	92.48	100.00	95.12	89.27	0.88	0.43	0.92	1.00	0.95	0.85	15.08
LOGR	83.02	48.40	86.36	98.21	92.22	75.00	0.84	0.33	0.88	1.00	0.95	0.77	17.55
KNN	90.22	70.68	91.15	99.06	94.20	87.76	0.92	0.39	0.96	1.00	0.97	0.91	9.89
SVC-LIN	71.21	14.16	70.49	98.51	78.54	63.85	0.63	0.38	0.60	1.00	0.69	0.52	27.60
SVC-POLY	74.08	34.51	73.41	98.81	80.00	67.31	0.64	0.17	0.62	1.00	0.70	0.54	26.18
SVC-RBF	73.68	19.47	73.41	97.91	79.50	67.32	0.65	0.22	0.62	1.00	0.73	0.54	25.38
MLP-LNF	85.16	42.72	91.15	99.40	94.97	82.77	0.82	0.18	0.93	1.00	0.96	0.76	15.93
MLP-SG	78.57	41.09	85.37	98.58	92.49	63.39	0.74	0.08	0.82	1.00	0.95	0.51	22.33
MLP-ADAM	86.90	45.77	91.15	99.30	94.63	84.64	0.85	0.18	0.94	1.00	0.98	0.79	12.73
BAG	89.46	73.37	90.25	98.58	93.83	86.87	0.92	0.29	0.96	1.00	0.98	0.90	7.78
RF	93.49	81.79	94.39	100.00	96.43	90.98	0.94	0.31	0.97	1.00	0.99	0.94	5.78
EXTR	94.28	81.68	95.07	99.76	96.90	92.10	0.94	0.35	0.98	1.00	0.99	0.95	5.35
AdaB	90.16	68.32	91.46	99.76	94.69	87.17	0.91	0.37	0.95	1.00	0.98	0.88	9.81
GraB	89.95	67.50	91.34	99.53	94.39	87.12	0.88	0.05	0.95	1.00	0.98	0.85	11.29
DL1	83.23	44.44	86.59	99.06	92.92	75.26	0.83	0.20	0.88	1.00	0.96	0.75	18.40
DL2	84.74	43.46	88.96	99.10	93.90	77.65	0.84	0.19	0.89	1.00	0.96	0.78	16.46
DL3	84.51	43.21	89.38	99.06	93.88	77.59	0.84	0.19	0.89	1.00	0.96	0.77	17.22
DL4	82.19	43.21	88.52	99.10	93.81	73.44	0.81	0.19	0.87	1.00	0.95	0.74	19.87
DL5	81.18	42.96	86.30	99.53	93.34	70.83	0.82	0.27	0.87	1.00	0.96	0.74	18.12
DL6	80.98	43.66	85.12	99.29	93.41	71.31	0.82	0.21	0.86	1.00	0.96	0.74	18.45
WELM-SIG	73.57	31.86	75.55	96.23	84.15	64.54	0.78	0.31	0.82	1.00	0.91	0.67	22.43
WELM-SIN	75.51	30.97	77.80	98.11	86.01	66.05	0.80	0.33	0.84	1.00	0.93	0.72	19.25
WELM-RBS	75.04	20.35	77.76	97.88	85.84	65.59	0.80	0.32	0.84	1.00	0.93	0.71	19.47
WELM-TRBS	75.56	20.35	77.84	98.11	86.34	66.09	0.81	0.34	0.85	1.00	0.94	0.71	18.09
LSSVM-LIN	86.51	55.20	89.38	100.00	94.25	79.78	0.89	0.54	0.93	1.00	0.98	0.82	9.85
LSSVM-Poly	92.21	58.91	93.89	100.00	97.78	89.36	0.95	0.63	0.98	1.00	1.00	0.93	3.87
LSSVM-RBF	98.23	85.15	99.06	100.00	99.51	97.80	0.99	0.74	1.00	1.00	1.00	1.00	1.18
ELM-LIN	77.30	49.12	80.76	98.35	89.38	66.34	0.70	0.33	0.71	0.98	0.84	0.50	27.26
ELM-RBF	79.91	49.27	83.49	98.11	90.71	71.32	0.72	0.49	0.75	0.98	0.86	0.53	26.08
ELM-Poly	81.89	49.27	86.29	98.82	92.48	73.79	0.75	0.48	0.78	0.99	0.89	0.62	24.08

Table 16. Descriptive Statistics of employed eight classifiers.

Fig. 10. Statistical test results: AUC: Classification Techniques.

MultinomialNB():{$'alpha'$: 1.0,$'class_prior'$: $None$,$'fit_prior'$: $True$,$'force_alpha'$: $True$}
BernoulliNB():{$'alpha'$: 1.0,$'binarize'$: 0.0,$'class_prior'$: $None$,$'fit_prior'$: $True$,$'force_alpha'$: $True$}
GaussianNB():{$'priors'$: $None$,$'var_smoothing'$: $1e-09$}
DecisionTreeClassifier():{$'ccp_alpha'$: 0.0,$'criterion'$:$'gini'$,$'min_samples_leaf'$: 1,$'min_samples_split'$: 2,$'min_weight_fraction_leaf'$: 0.0,$'splitter'$:$'best'$}
LogisticRegression():{$'C'$: 1.0,$'fit_intercept'$: $True$,$'intercept_scaling'$: 1,$'max_iter'$: 100,$'n_jobs'$: $None$,$'penalty'$:$'l2'$,$'solver'$:$'lbfgs'$,$'tol'$: 0.0001}
KNeighborsClassifier():{$'algorithm'$:$'auto'$,$'leaf_size'$: 30,$'metric'$:$'minkowski'$,$'n_jobs'$: $None$,$'n_neighbors'$: 5,$'p'$: 2,$'weights'$:$'uniform'$}
SVC(kernel='linear'):{$'C'$: 1.0,$'cache_size'$: 200,$'decision_function_shape'$:$'ovr'$,$'degree'$: 3,$'gamma'$:$'scale'$,$'kernel'$:$'linear'$,$'probability'$: $True$,$'tol'$: 0.001}
SVC(kernel='poly'):{$'C'$: 1.0,$'cache_size'$: 200,$'decision_function_shape'$:$'ovr'$,$'degree'$: 3,$'gamma'$:$'scale'$,$'kernel'$:$'poly'$,$'probability'$: $True$,$'tol'$: 0.001}
SVC(kernel='rbf'):{$'C'$: 1.0,$'cache_size'$: 200,$'decision_function_shape'$:$'ovr'$,$'degree'$: 3,$'gamma'$:$'scale'$,$'kernel'$:$'rbf'$,$'probability'$: $True$,$'tol'$: 0.001}
LSSVC(kernel='linear'):{$'C'$: 1.0,$'cache_size'$: 200,$'decision_function_shape'$:$'ovr'$,$'degree'$: 3,$'gamma'$:$'scale'$,$'kernel'$:$'linear'$,$'probability'$: $True$,$'tol'$: 0.001}
LSSVC(kernel='poly'):{$'C'$: 1.0,$'cache_size'$: 200,$'decision_function_shape'$:$'ovr'$,$'degree'$: 3,$'gamma'$:$'scale'$,$'kernel'$:$'poly'$,$'probability'$: $True$,$'tol'$: 0.001}
LSSVC(kernel='rbf'):{$'C'$: 1.0,$'cache_size'$: 200,$'decision_function_shape'$:$'ovr'$,$'degree'$: 3,$'gamma'$:$'scale'$,$'kernel'$:$'rbf'$,$'probability'$: $True$,$'tol'$: 0.001}
MLPClassifier(solver='lbfgs'):{$'activation'$:$'relu'$,$'alpha'$: 0.0001,$'batch_size'$:$'auto'$,$'beta_1$: 0.9,$'beta_2$: 0.999,$'early_stopping'$: $False$,$'epsilon'$: $1e-08$,$'hidden_layer_sizes'$: (320, 2),$'learning_rate_init'$: 0.001,$'max_fun'$: 15000,$'max_iter'$: 1500,$'momentum'$: 0.9}
MLPClassifier(solver='sgd'):{$'activation'$:$'relu'$,$'alpha'$: 0.0001,$'batch_size'$:$'auto'$,$'beta_1$: 0.9,$'beta_2$: 0.999,$'early_stopping'$: $False$,$'epsilon'$: $1e-08$,$'hidden_layer_sizes'$: (320, 2),$'learning_rate_init'$: 0.001,$'max_fun'$: 15000,$'max_iter'$: 1500,$'momentum'$: 0.9}
MLPClassifier(solver='adam'):{$'activation'$:$'relu'$,$'alpha'$: 0.0001,$'batch_size'$:$'auto'$,$'beta_1$: 0.9,$'beta_2$: 0.999,$'early_stopping'$: $False$,$'epsilon'$: $1e-08$,$'hidden_layer_sizes'$: (320, 2),$'learning_rate_init'$: 0.001,$'max_fun'$: 15000,$'max_iter'$: 1500,$'momentum'$: 0.9}
BaggingClassifier():{$'bootstrap'$: $True$,$'bootstrap_features'$: $False$,$'estimator_alpha'$: 1.0,$'estimator_fit_prior'$: $True$,$'estimator_force_alpha'$: $True$,$'estimator'$: $MultinomialNB()$,$'max_features'$: 0.5,$'max_samples'$: 0.5,$'n_estimators'$: 10}
RandomForestClassifier(n_estimators=10):{$'bootstrap'$: $True$,$'ccp_alpha'$: 0.0,$'criterion'$:$'gini'$,$'max_features'$:$'sqrt'$,$'min_impurity_decrease'$: 0.0,$'min_samples_leaf'$: 1,$'min_samples_split'$: 2,$'min_weight_fraction_leaf'$: 0.0,$'n_estimators'$: 10}
ExtraTreesClassifier(n_estimators=10, random_state=0){$'bootstrap'$: $False$,$'ccp_alpha'$: 0.0,$'criterion'$:$'gini'$,$'max_features'$:$'sqrt'$,$'min_impurity_decrease'$: 0.0,$'min_samples_leaf'$: 1,$'min_samples_split'$: 2,$'min_weight_fraction_leaf'$: 0.0,$'monotonic_cst'$: $None$,$'n_estimators'$: 10}
GradientBoostingClassifier(learning_rate=1.0,max_depth=1,n_estimators=10, random_state=0){$'ccp_alpha'$: 0.0,$'criterion'$:$'friedman_mse'$,$'learning_rate'$: 1.0,$'loss'$:$'log_loss'$,$'max_depth'$: 1,$'min_impurity_decrease'$: 0.0,$'min_samples_leaf'$: 1,$'min_samples_split'$: 2,$'min_weight_fraction_leaf'$: 0.0,$'n_estimators'$: 10,$'n_iter_no_change'$: $None$,$'random_state'$: 0,$'subsample'$: 1.0,$'tol'$: 0.0001,$'validation_fraction'$: 0.1,$'verbose'$: 0}
ELM(kernel='linear'):{$'C'$: 1.0,$'Kernel_para'$: 2,$'NumberofHiddenNeurons'$: 320,$'degree'$: 3,$'Regularization_coefficient'$: 1,$'kernel'$:$'linear'$}
ELM(kernel='poly'):{$'C'$: 1.0,$'Kernel_para'$: 2,$'NumberofHiddenNeurons'$: 320,$'degree'$: 3,$'Regularization_coefficient'$: 1,$'kernel'$:$'poly'$}
ELM(kernel='rbf'):{$'C'$: 1.0,$'Kernel_para'$: 2,$'NumberofHiddenNeurons'$: 320,$'degree'$: 3,$'Regularization_coefficient'$: 1,$'kernel'$:$'rbf'$}
AdaBoostClassifier(n_estimators=10){$'algorithm'$:$'SAMME.R'$,$'estimator'$: $None$,$'learning_rate'$: 1.0,$'n_estimators'$: 10,$'random_state'$: $None$}
DL1 < Sequentialname=sequential_22, built=True > {$'name'$:$'adam'$,$'learning_rate'$: 0.0010000000474974513,$'weight_decay'$: $None$,$'clipnorm'$: $None$,$'global_clipnorm'$: $None$,$'clipvalue'$: $None$,$'use_ema'$: $False$,$'ema_momentum'$: 0.99,$'ema_overwrite_frequency'$: $None$,$'loss_scale_factor'$: $None$,$'gradient_accumulation_steps'$: $None$,$'beta_1$: 0.9,$'beta_2$: 0.999,$'epsilon'$: $1e-07$,$'amsgrad'$: $False$}
DL2 < Sequential name=sequential_23, built=True > {$'name'$:$'adam'$,$'learning_rate'$: 0.0010000000474974513,$'weight_decay'$: $None$,$'clipnorm'$: $None$,$'global_clipnorm'$: $None$,$'clipvalue'$: $None$,$'use_ema'$: $False$,$'ema_momentum'$: 0.99,$'ema_overwrite_frequency'$: $None$,$'loss_scale_factor'$: $None$,$'gradient_accumulation_steps'$: $None$,$'beta_1$: 0.9,$'beta_2$: 0.999,$'epsilon'$: $1e-07$,$'amsgrad'$: $False$}
DL3 < Sequential name=sequential_24, built=True > {$'name'$:$'adam'$,$'learning_rate'$: 0.0010000000474974513,$'weight_decay'$: $None$,$'clipnorm'$: $None$,$'global_clipnorm'$: $None$,$'clipvalue'$: $None$,$'use_ema'$: $False$,$'ema_momentum'$: 0.99,$'ema_overwrite_frequency'$: $None$,$'loss_scale_factor'$: $None$,$'gradient_accumulation_steps'$: $None$,$'beta_1$: 0.9,$'beta_2$: 0.999,$'epsilon'$: $1e-07$,$'amsgrad'$: $False$}

Table 17. Hyper-parameter values for the ML models.

Data availability

The data used in this paper is available at https://github.com/ouniali/WSantipatterns. The processed data will be made available on request.

Appendix

See Table 17.

Received: 9 August 2024; Accepted: 10 January 2025
Published online: 12 February 2025

References

1. Kral, J. & Zemlicka, M. The most important service-oriented antipatterns. In *International Conference on Software Engineering Advances (ICSEA 2007)* pp. 29–29. IEEE (2007).
2. Travassos, G., Shull, F., Fredericks, M. & Basili, V. R. Detecting defects in object-oriented designs: using reading techniques to increase software quality. *ACM Sigplan Notices* **34**(10), 47–56 (1999).
3. Marinescu, R. Detection strategies: Metrics-based rules for detecting design flaws. In *20th IEEE International Conference on Software Maintenance, 2004. Proceedings*, pp. 350–359. IEEE (2004).

4. Munro, M. J. Product metrics for automatic identification of "bad smell" design problems in java source-code. In *11th IEEE International Software Metrics Symposium (METRICS'05)*, pp. 15–15. IEEE (2005).
5. Ciupke, O. Automatic detection of design problems in object-oriented reengineering. In *Proceedings of Technology of Object-oriented Languages and Systems-TOOLS 30 (Cat. No. PR00278)*, pp. 18–32. IEEE (1999).
6. Simon, F., Steinbruckner, F. & Lewerentz, C. Metrics based refactoring. In *Proceedings fifth European conference on software maintenance and reengineering*, pp. 30–38. IEEE (2001).
7. Ananda Rao, A. & Reddy, K. N. Detecting bad smells in object oriented design using design change propagation probability matrix 1 (2007).
8. Khomh, F., Vaucher, S., Guéhéneuc, Y.-G. & Sahraoui, H. Bdtex: A gqm-based bayesian approach for the detection of antipatterns. *J. Syst. Softw.* **84**(4), 559–572 (2011).
9. Moha, N., Guéhéneuc, Y.-G., Duchien, L. & Le Meur, A.-F. Decor: A method for the specification and detection of code and design smells. *IEEE Trans. Softw. Eng.* **36**(1), 20–36 (2009).
10. Chidamber, S. R. & Kemerer, C. F. A metrics suite for object oriented design. *IEEE Trans. Softw. Eng.* **20**(6), 476–493 (1994).
11. Hemanta Kumar Bhuyan and Vinayakumar Ravi. Analysis of subfeature for classification in data mining. *IEEE Trans. Eng. Manag.* **70**(8), 2732–2746 (2021).
12. Hemanta Kumar Bhuyan and Narendra Kumar Kamila. Privacy preserving sub-feature selection based on fuzzy probabilities. *Clust. Comput.* **17**(4), 1383–1399 (2014).
13. Bhuyan, H. K., Saikiran, M., Tripathy, M. & Ravi, V. Wide-ranging approach-based feature selection for classification. *Multimedia Tools Appl.* **82**(15), 23277–23304 (2023).
14. Vasilescu, B., Serebrenik, A. & Van den Brand, M. By no means: A study on aggregating software metrics. In *Proceedings of the 2nd International Workshop on Emerging Trends in Software Metrics*, pp. 23–26 (2011).
15. Fernández, A., Garcia, S., Herrera, F. & Chawla, N. V. Smote for learning from imbalanced data: Progress and challenges, marking the 15-year anniversary. *J. Artif. Intell. Res.* **61**, 863–905 (2018).
16. Xiaowei, G., Angelov, P. P. & Soares, E. A. A self-adaptive synthetic over-sampling technique for imbalanced classification. *Int. J. Intell. Syst.* **35**(6), 923–943 (2020).
17. Tang, Y., Zhang, Y.-Q., Chawla, N. V. & Krasser, S. Svms modeling for highly imbalanced classification. *IEEE Trans. Syst. Man Cybern. Part B (Cybernetics)* **39**(1), 281–288 (2008).
18. Ghorbani, R. & Ghousi, R. Comparing different resampling methods in predicting studentsâ€™ performance using machine learning techniques. *IEEE Access* **8**, 67899–67911 (2020).
19. Zhiquan, H., Wang, L., Qi, L., Li, Y. & Yang, W. A novel wireless network intrusion detection method based on adaptive synthetic sampling and an improved convolutional neural network. *IEEE Access* **8**, 195741–195751 (2020).
20. He, H., Bai, Y., Garcia, E. A & Li, S. Adasyn: Adaptive synthetic sampling approach for imbalanced learning. In *2008 IEEE iNternational Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence)*, pp. 1322–1328. IEEE (2008).
21. Utami, E., Oyong, I., Raharjo, S., Hartanto, A. D. & Adi, S.. Supervised learning and resampling techniques on disc personality classification using twitter information in Bahasa Indonesia. *Appl. Comput. Inform.* (2021).
22. Vandana, C. P. & Chikkamannur, A. A. Feature selection: An empirical study. *Int. J. Eng. Trends Technol.* **69**(2), 165–170 (2021).
23. Di Mauro, M., Galatro, G., Fortino, G. & Liotta, A. Supervised feature selection techniques in network intrusion detection: A critical review. *Eng. Appl. Artif. Intell.* **101**, 104216 (2021).
24. Dogra, V., Singh, A., Verma, S., Jhanjhi, N. Z & Talib, M. N. et al. Understanding of data preprocessing for dimensionality reduction using feature selection techniques in text classification. In *Intelligent computing and innovation on data science*, pp. 455–464. Springer (2021).
25. Zebari, R., Abdulazeez, A., Zeebaree, D., Zebari, D. & Saeed, J. A comprehensive review of dimensionality reduction techniques for feature selection and feature extraction. *J. Appl. Sci. Technol. Trends* **1**(1), 56–70 (2020).
26. Stiawan, D. et al. Cicids-2017 dataset feature analysis with information gain for anomaly detection. *IEEE Access* **8**, 132911–132921 (2020).
27. Hasdyna, N., Sianipar, B., & Zamzami, E. M. Improving the performance of k-nearest neighbor algorithm by reducing the attributes of dataset using gain ratio. J. Phys.: Conf. Ser., vol. 1566, p. 012090. IOP Publishing (2020).
28. Al Sayaydeha, Osama Nayel & Mohammad, Mohammad Falah. Diagnosis of the parkinson disease using enhanced fuzzy min-max neural network and oner attribute evaluation method. In *2019 International Conference on Advanced Science and Engineering (ICOASE)*, pp. 64–69. IEEE (2019).
29. Karamizadeh, S., Abdullah, S. M., Manaf, A. A., Zamani, M. & Hooman, A. An overview of principal component analysis. *J. Signal Inf. Process.*, **4** (2020).
30. Meng, X.-L., Rosenthal, R. & Rubin, D. B. Comparing correlated correlation coefficients. *Psychol. Bull.* **111**(1), 172 (1992).
31. Xu, S., Zhang, Z., Wang, D., Hu, J., Duan, X. & Zhu, T. Cardiovascular risk prediction method based on cfs subset evaluation and random forest classification framework. In *2017 IEEE 2nd International Conference on Big Data Analysis (ICBDA)*, pp. 228–232. IEEE (2017).
32. Too, J. and Abdullah, A. R. A new and fast rival genetic algorithm for feature selection. *J. Supercomput.* **77**(3), 2844–2874 (2021).
33. Romano, M., Contu, G., Mola, F. & Conversano, C. Threshold-based naïve bayes classifier. *Adv. Data Anal. Classification*, pp. 1–37 (2023).
34. Prasanta Kumar Dey. Project risk management: A combined analytic hierarchy process and decision tree approach. *Cost Eng.* **44**(3), 13–27 (2002).
35. Niu, L. A review of the application of logistic regression in educational research: Common issues, implications, and suggestions. *Educ. Rev.* **72**(1), 41–67 (2020).
36. Alam, S., Sonbhadra, S. K., Agarwal, S. & Nagabhushan, P. One-class support vector classifiers: A survey. *Knowl.-Based Syst.* **196**, 105754 (2020).
37. Leong, W. C., Bahadori, A., Zhang, J. & Ahmad, Z. Prediction of water quality index (wqi) using support vector machine (svm) and least square-support vector machine (ls-svm). *Int. J. River Basin Manag.* **19**(2), 149–156 (2021).
38. Wang, J., Siyuan, L., Wang, S.-H. & Zhang, Y.-D. A review on extreme learning machine. *Multimedia Tools Appl.* **81**(29), 41611–41660 (2022).
39. Liu, Z., Jin, W. & Ying, M. Variances-constrained weighted extreme learning machine for imbalanced classification. *Neurocomputing* **403**, 45–52 (2020).
40. Heidari, A. A., Faris, H., Mirjalili, S., Aljarah, I. & Mafarja, M. Ant lion optimizer: theory, literature review, and application in multi-layer perceptron neural networks. *Nature-Inspired Optimizers: Theories, Literature Reviews and Applications*, pp. 23–46 (2020).
41. Boateng, E. Y., Otoo, J. & Abaye, D. A. Basic tenets of classification algorithms k-nearest-neighbor, support vector machine, random forest and neural network: a review. *J. Data Anal. Inf. Process.* **8**(4), 341–357 (2020).
42. Khan, Z. et al. Optimal trees selection for classification via out-of-bag assessment and sub-bagging. *IEEE Access* **9**, 28591–28607 (2021).
43. Ahangari, S., Jeihani, M., Anam Ardeshiri, Md., Rahman, M. & Dehzangi, A. Enhancing the performance of a model to predict driving distraction with the random forest classifier. *Transp. Res. Rec.* **2675**(11), 612–622 (2021).

44. Abubaker, H., Ali, A., Shamsuddin, S. M. & Hassan, S. Exploring permissions in android applications using ensemble-based extra tree feature selection. *Indonesian J. Electr. Eng. Comput. Sci.* **19**(1), 543–552 (2020).
45. Malhotra, R. & Jain, J. Handling imbalanced data using ensemble learning in software defect prediction. In *2020 10th International Conference on Cloud Computing, Data Science and Engineering (Confluence)*, pp. 300–304. IEEE (2020).
46. Nguyen, H. & Hoang, N.-D. Computer vision-based classification of concrete spall severity using metaheuristic-optimized extreme gradient boosting machine and deep convolutional neural network. *Autom. Constr.* **140**, 104371 (2022).
47. Sarker, I. H. Deep learning: A comprehensive overview on techniques, taxonomy, applications and research directions. *SN Comput. Sci.* **2**(6), 420 (2021).

Author contributions
LK and ST conceptualize the topic. ST, LK, LBM, SM and AK are involved in Methodology, investigation, and validation. LK, ST and SM supervised the whole work. All authors reviewed the manuscript.

Declarations

Competing interests
The authors declare no competing interests.

Ethical and informed consent for data used:
No ethical approval and consent are required based on the following. (a) This article does not contain any studies with animals performed by any of the authors. (b) This article does not contain any studies with human participants or animals performed by any of the authors. (c) This article does not use any figure or table from any source.

Additional information
Correspondence and requests for materials should be addressed to L.K. or S.M.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.