Cluster Computing
https://doi.org/10.1007/s10586-023-04132-5

Boosting the visibility of services in microservice architecture

Ahmet Vedat Tokmak[1] · Akhan Akbulut[1] · Cagatay Catal[2]

Received: 29 January 2023 / Revised: 19 August 2023 / Accepted: 27 August 2023

Abstract
Monolithic software architectures are no longer sufficient for the highly complex software-intensive systems, which modern society depends on. Service Oriented Architecture (SOA) surpassed monolithic architecture due to its reusability, platform independency, ease of maintenance, and scalability. Recent SOA implementations made use of cloud-native architectural approaches such as microservice architecture, which has resulted in a new challenge: the discovery difficulties of services. One way to dynamically discover and route traffic to service instances is to use a service discovery tool to locate the Internet Protocol (IP) address and port number of a microservice. In the event that replicated microservice instances are found to provide the same function, it is crucial to select the right microservice that provides the best overall experience for the end-user. Parameters including success rate, efficiency, delay time, and response time play a vital role in establishing a microservice's Quality of Service (QoS). These assessments can be performed by means of a live health-check service, or, alternatively, by making a prediction of the current state of affairs with the application of machine learning-based approaches. In this research, we evaluate the performance of several classification algorithms for estimating the quality of microservices using the QWS dataset containing traffic data of 2505 microservices. Our research also analyzed the boosting algorithms, namely Gradient Boost, XGBoost, LightGBM, and CatBoost to improve the overall performance. We utilized parameter optimization techniques, namely Grid Search, Random Search, Bayes Search, Halvin Grid Search, and Halvin Random Search to fine-tune the hyperparameters of our classifier models. Experimental results demonstrated that the CatBoost algorithm achieved the highest level of accuracy (90.42%) in predicting microservice quality.

Keywords Service discovery · Microservice architecture · Boosting algorithms · CatBoost · LightGBM · XGBoost · Gradient increase

1 Introduction

Currently, the widespread use of information technologies permits the expansion of applications' capacities. Although developing applications using monolithic structures may seem advantageous for small-scale applications, this strategy creates scalability issues for medium and large-scale systems. Service Oriented Architecture (SOA) is a strategy for decomposing systems into separate threads devoted to distinct activities as opposed to a single component. SOA is becoming increasingly popular because it is language and platform agnostic and more scalable. The architecture of microservices has been depicted in Fig. 1. It is presently evolving to incorporate microservices to address challenges associated with monolithic architecture's limitations [1]. Microservice architectures have been widely adopted and utilized in various industries for over a decade. Their extensive use demonstrated their effectiveness in improving the scalability, maintainability, and flexibility of systems. The widespread implementation of these architectures highlights their relevance in modern software development practices. Savings in both time and money

✉ Cagatay Catal
ccatal@qu.edu.qa

Ahmet Vedat Tokmak
2000005655@stu.iku.edu.tr

Akhan Akbulut
a.akbulut@iku.edu.tr

[1] Department of Computer Engineering, Istanbul Kültür University, 34536 Istanbul, Turkey

[2] Department of Computer Science and Engineering, Qatar University, Doha 2713, Qatar

Published online: 18 September 2023

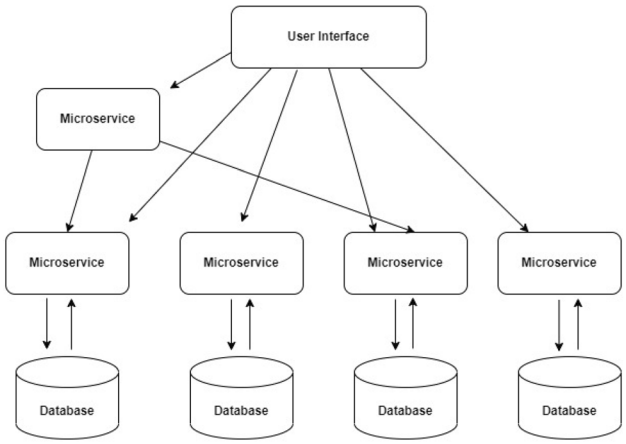

Fig. 1 Microservice architecture

can be achieved since services can be reused across different programs. Because each service is installed separately, the other services are not affected adversely in the case of a failure of a particular service. Also, it has made maintenance and troubleshooting much simpler. During the service discovery phase, it is crucial to perform an accurate quality prediction to select the best service. The aim of this research is to perform a highly accurate prediction of the quality of service in the microservice ecosystem.

As a result of advancements in the software industry, the number of microservices in use is accelerating. Large applications are composed of numerous microservices that perform distinct functions; therefore, it is faster and more effective to select the appropriate language for each microservice than to write the entire application in a single language. Developers favor microservices because to this characteristic. The dynamic nature of massive Internet networks and web services, however, makes service discovery problematic. Service discovery is the most fundamental microservices challenge. In most cases, when attempting to discover a service, you will want to concentrate on these three ideas: first, the service's IP and port number must be discovered; next, the service's health must be verified; and finally, load balancing must be used to route the request to the best service available. Classification and clustering of comparable services are two of the most popular approaches used to optimize the service discovery process and locate the best service quickly [2].

Failure to effectively operate the service discovery process will prevent the microservice ecosystem from achieving the intended level of efficiency from its services, or prevent the services from being utilized at all. Obtaining the service's location information is the initial challenge encountered during service discovery. Due to the microservice ecosystem's dynamic structure, the geographical information of the services is continually changing, making it difficult to discover them. If access to the location data of the services is denied, the services cannot be utilized. There are also other services with comparable functions, which presents a dilemma. In this situation, it is crucial to understand the quality of the services in order to choose the finest one. If service quality cannot be correctly predicted, it will be difficult to choose the best decision. If the finest service is not chosen, the anticipated benefit will not be realized. This study centered on determining the optimal method for estimating service quality.

Examining the literature on microservices discovery techniques reveals that the most researched topic is the preference between services that execute comparable functions. An online store might, for instance, employ several distinct payment gateways. The primary issue is that it is unknown to the user which of the available services they will be referred to during the actual payment process. When this happens, we need to identify the various services that accomplish the same goal and rank them according to quality standards so that we may choose the most advantageous one. The efficiency and precision of the estimation demonstrate the efficacy of the strategy employed in service discovery. Extensive usage of various classification strategies has yielded successful outcomes in prior research. Logistic Regression, Random Forest (RF), Support Vector Machine (SVM) [3], Decision Tree (DT) [4] and Naive Bayes [5] are among the most popular algorithms. The ultimate goal of these many approaches and algorithms is to properly categorize services and provide reliable estimates of their quality, but these goals might vary greatly.

Boosting algorithms are, in essence, a form of community-based learning similar to the Random Forest algorithm. In Boosting algorithms, unlike the Random Forest technique, classifiers create new predictions by considering the prediction of the previous classifier, and this process continues until no further model improvement is possible. As a result of these characteristics, Boosting algorithms provided improved solutions to classification problems. Using Boosting algorithms such as Gradient Boosting, XGBoost, LightGBM, and CatBoost, more accurate estimates of service quality were obtained during the service discovery procedure. The most essential characteristic that gives these algorithms their strength is their propensity to reduce past estimation errors.

The problem addressed in this research is the accurate estimation of Quality of Service (QoS) parameters in the context of microservices. It is critical to achieve optimal performance and enhance the user experience by predicting these parameters accurately. To tackle this problem, we performed a series of experiments to evaluate various machine learning classification algorithms. Among them, our findings demonstrate that the CatBoost algorithm

outperforms others in predicting the QoS parameters of microservices.

The main contributions of this study are fourfold, which are listed as follows:

- By leveraging the QWS dataset, we developed a machine learning-based prediction model that surpasses previous research studies in accurately predicting service quality. One key advantage of our prediction model lies in its exceptional accuracy, significantly improving the precision of service quality estimation.
- Experiments with various Boosting algorithms achieved remarkable results.
- Parameter selection and hyperparameter optimization were both performed using metaheuristics and helped during the experiments.
- Various metrics were utilized to confirm the efficacy of boosting algorithms in predicting service quality.

The remainder of the paper is structured as follows: Studies on web service discovery are discussed in Sect. 2. The third section describes the approach employed. In Sect. 4, we present a synopsis of the experimental results. Discussion of the study is included in Sect. 5. Finally, Sect. 6 concludes the paper.

2 Related work

Researchers have been aiming for the most precise service possible by comparing and contrasting available techniques. Accessing the most accurate service in a short time is the main challenge in the microservice architecture, which is gaining popularity due to its benefits. Classification and clustering of services that perform comparable activities is the most popular approach according to some researchers [2]. Evaluation of the quality of web services is of great importance in selecting a web service for an application [6]. Quality features of web services are classified using classifiers such as BPNN, PNN, GMDH, CART, J48, TreeNet and SVM to get the best results for services with similar features [7]. Researchers try to produce solutions at the stage of reaching the most appropriate service by using different approaches for classification and clustering. The approach of classifying web services through dynamically generated recognition modules according to quality parameters provides a solution to the selection problem by using less complex tasks [8].

The fundamental challenges encountered in the classification of web services often stem from inadequacies in the description texts associated with these services. To enhance this classification, various methods have been employed. One such method involves the use of Recurrent Convolutional Neural Networks (RCNN) [9]. RCNN effectively utilizes web service descriptions and the actions within these descriptions to attain successful outcomes. This approach, centered around text matching, has elevated the performance of discovery processes. Additionally, methods such as term frequency–inverse document frequency (TF–IDF), Word2Vec, and ELMo, have been successfully applied in capturing the word frequency of each keyword, along with static and dynamic contextual features [10]. Another method aimed at reducing web service access times and enhancing the efficiency of discovery is the Concept Lattice. In this approach, relevant web services are extracted based on an ontology derived from user queries [11]. The ID-LSTM model also emerges as a proposed approach for web service classification. This method employs reinforcement learning to filter out unnecessary information and retain pertinent details. Subsequently, the classification network evaluates distilled information and updates itself based on classification outcomes [12]. The Density Clustering Services method, which combines Latent Semantic Analysis (LSA) and IO-MATCHING semantic matching, measures semantic similarity, resulting in highly accurate clustering of web services from a semantic perspective [13]. Furthermore, automatic classification of a web service into an appropriate category based on the provided descriptions can augment the accuracy and ease of discovery. This can be achieved through numerical–statistical methods such as Term Frequency–Inverse Document Frequency (TF–IDF) and the Multinomial Naive Bayes classification algorithm [14].

The selection of services to be utilized may necessitate alignment with expressions found within user queries. One of the methods employed to establish semantic connections between user queries and service descriptions is the Bi-LSTM (Bidirectional Long Short-Term Memory), which captures the textual meanings of sentences [15]. By utilizing the attention-based Bi-LSTM model, focus can be directed towards subtle factors within web services, such as the varying significance of different words and sequential semantic relationships between words [16]. Specifically, BiLSTM is employed to automatically learn keyword feature representations of web services. These attributes can be effectively utilized by a softmax neural network for the final web service classification [17]. Graph Neural Network (GNN)-based service classification emerges as another effective approach recommended for service discovery. This method, based on graph neural networks' information distillation, can construct a service relationship network by first utilizing information from API service descriptions to extract feature vectors [18]. Another method utilized to extract more features that impact the accuracy of web service classification is the Residual Attention Graph Convolution Network (RAGCN) model. This model integrates an attention mechanism into graph

convolution networks and employs residual learning to enhance the depth of the model, facilitating the extraction of additional features [19]. Furthermore, for alleviating issues of data sparsity and noise, embedding words using Word2Vec from web service descriptions enhances the success of service classification [20]. Mapping services in a vector space by analyzing semantic meaning and context enables effective web service clustering [21]. Placing information about services into a knowledge graph mitigates the impact of data sparsity and reveals deep relationships between services, thereby enhancing the accuracy of service discovery [22]. The method known as Semantic Information Extension for service information graphs facilitates the discovery of relationships between services and aids in classifying evolving services based on service definitions [23]. A novel deep neural network with the Co-Attentive Representation Learning (CARL-Net) mechanism effectively classifies services by learning interconnected features of services [24]. The Att-RTM model, aimed at representing web services with low-dimensional vectors, is a functional attention-based probabilistic model that extracts hidden topics of services while emphasizing words relevant to functionality, thus enhancing prediction accuracy [25].

One of the challenging aspects is organizing and classifying web services based on their syntax while disregarding the content that the requested service carries. The Cloud-Based Classification Methodology (CBCM) presents an approach for classifying semantic web services. The methodology comprises three main modules: scanning, filtering, and conceptualization processes. This method enhances the intelligent performance of the classifier by focusing not only on the texts but also on the meanings of concepts [26]. The challenge intensifies when end-users search for relevant cloud services from a diverse range of heterogeneous web resources. Understanding the Quality of Experience (QoEgure) of end-users and determining whether consumer reviews are positive, negative, or neutral provides potential consumers with effective and efficient service discovery [27]. In a multi-cloud environment, a semantic-based approach called S-CogSD allows the formal definition of certain cognitive functions. It employs a cognitive ontology to understand users' requests and categorizes cognitive services into functionally similar service groups. This approach facilitates better comprehension of user requests and the creation of equivalent service groups [28].

Although it is the most common method to use the attributes of the services to facilitate the selection of services, this method uses many different techniques. The Tabu Search (Tabu Search-TS) algorithm uses adaptive memory as well as responsive discovery. It starts similar to the neighbor search and progresses iteratively from one point to another until it meets a specified criterion [29]. Naive Bayes is a widely used classification method based on Bayesian theory. A Bayesian classifier assumes that a particular feature of a class has nothing to do with other features [30]. Random Forest is a method of classification and regression based on the assembly of a large number of decision trees. This algorithm enables us to create different models and classify each of the created decision trees by training them on a different observation [31]. The K-Means algorithm determines the quality of the cluster by calculating the square error between the average of a cluster and the points in the cluster. The purpose of the algorithm is to minimize the sum of the squared error on all K sets [32]. Support Vector Machine is a machine learning technique used in the field of data mining. The algorithm is defined as a supervised learning algorithm that solves the problem of linear and nonlinear binary classification. As can be seen, although all of the studies are using different techniques and algorithms, it is the best service classification according to the attributes. Other approaches used in grouping and sequencing of services are artificial neural networks [33], artificial bee colony algorithm [34] and approaches where different approaches are used [35] as hybrids. In all approaches, the common goal is to group and sort the services in order to reach the most accurate service in the fastest way. Table 1 lists the classifiers preferred by the researchers and the accuracy rates they obtained in the studies on service quality estimation in the microservice ecosystem.

3 Methodology

This section explains the parameters utilized in the tests, the QWS dataset used in the studies, and the reasons why Boosting algorithms were chosen first and subsequently the classification algorithms selected for comparison. Finally, the procedures for parameter and hyperparameter selection and adjustment, as well as evaluation, are detailed.

Improved microservice quality prediction is advocated through the use of boosting methods. When different weights are applied to the dataset, a community of trees is generated, and it is from this group of trees that inferences are drawn using boosting methods. At the outset, all observations are given the same importance; but, as the tree community develops, the weightings are reshuffled. All of the trees in a Boosting algorithm are linked together, and each classifier is educated to consider the achievements of its predecessors. To improve the odds of picking the best target, Boosting is used. Popular boosting algorithms include Gradient Boosting, XGBoost, LightGBM, and CatBoost [36].

Table 1 Research on service quality estimation

Authors	Classifiers	Accuracy rates
Mohanty Ramakanta, Vadlamani Ravi and Manas Ranjan Patra (2010)	PNN (Probabilistic Neural Network)	89.99
	BPNN (Backpropagation Neural Network)	86.38
	GMDH (Group Data Processing Method)	89.75
	J48 (ID3 Decision Tree)	67.77
	TreeNet (Gradient Increasing Machine)	82.44
	Classification and Regression Trees	78.61
	SVM (Support Vector Machine)	60.55
A. Syed Mustafa and Y.S. Kumaraswamy (2014)	Naive Bayes	77.81
	C 4.5 (Improved version of ID3 Decision Tree)	79.45
	Random Forest	82.19
Ramakanta Mohanty, Aishwarya Priyadarshini and Medha Chippa (2019)	Naive Bayes	82.70
	KNN	80.29
	Random Forest	99.44
	SVM	98.07
	Decision Trees	99.72
	Linear SVM	85.43
Noor Al-Huda Hamed Olewy and Emir Kadhim Hadi (2021)	Random Forest	99.9
	SVM	98.6
	NN (Neural Network)	96.4
	Logistic Regression	92
Syed Mustafa and Kumara Swamy Y.S. (2015)	Multi-Layer Perceptron optimized with Tabu search	95.34
	Multi-Layer Perceptron-Levenberg–Marquardt	96.99
	Multi-Layer Perceptron Back Propagation	97.53
A. Banka, N. Juneja, A. Shrimal, S. Agrawal and L. Purohit (2019)	LMT	83.51
	Logistic Regression	84.34
	Naive Bayes	53.02
	Random Forest	80.22
A. Laachemi and D. Boughaci (2016)	Naive Bayes	81.31
	SVM	83.52
	Kstar	81.32
	Adaboost	60.44
A. Laachemi and D. Boughaci (2017)	Naive Bayes	81.87
	Naive Bayes + stochastic local search (SLS)	83.53

Gradient Boosting is a powerful machine learning technique introduced in 2001. In Gradient Boosting, the primary step involves constructing the initial decision tree. Subsequently, new trees are built by considering prediction errors. This process continues until no further improvement can be made in the model. It can be used for both regression and classification models [37]. XGBoost, developed by Tianqi Chen and Carlos Guestrin in 2016, is an optimized implementation of Gradient Boosting that focuses on parallel processing, tree pruning, handling missing values, and regularization to prevent overfitting. It is known as a regularized boosting technique as it reduces overfitting and enhances overall performance [38]. LightGBM, developed under the Microsoft DMTK (Distributed Machine Learning Toolkit) project in 2017, is an enhancement algorithm. It stands out due to its advantages such as high processing speed, the ability to handle large datasets, efficient memory usage, high prediction accuracy, support for parallel learning, and GPU training when compared to other boosting algorithms. CatBoost, an open-source machine learning algorithm based on Gradient Boosting, was developed by Yandex. Introduced in April

2017, it aims to enhance the performance of the Gradient Boosting algorithm and serves as an alternative to XGBoost and LightGBM. Its distinctive features include high learning speed, compatibility with numeric, categorical, and text data, GPU support, and visualization options [39]. Furthermore, proper tuning of hyperparameters is crucial for improving success in Boosting algorithms.

Literature investigation yielded positive findings, and hence five distinct classification algorithms were identified for comparison with the Boosting methods. [40]. Since Decision Tree is a classification method that creates a model in the form of a tree, such as Boosting algorithms, Random Forest has been preferred because it is a community learning method. Naive Bayes, another of these algorithms, was chosen because it was a probabilistic approach, while KNN, another algorithm, was preferred because it assigned the class to the most ideal class by comparing the unknown data with other data. SVM, which was the last algorithm selected, was included in the comparison algorithms because it had related learning algorithms that analyzed the data.

Our research compared the performance of various classification algorithms on the updated version of the publicly accessible QWS (Quality of Web Service) dataset [41]. The Quality of Web Service (QWS) dataset is a compilation of QWS metrics for 2505 operational Web services. Table 2 displays the QWS dataset's parameters alongside brief descriptions of each. By combining the first nine parameters, we arrived at the WsRF (Web Services Resource Framework) [42] parameter. Based on the WsRF's standard quality rating, the QWS dataset's categorization parameter was built up as follows: 1, Platinum (High Quality), 2, Gold (Medium Quality), 3, Silver (Low Quality), 4, Bronze (Very Low Quality). The dataset includes 412 records for first-class service, 1268 for second-class, 617 for third-, and 208 for fourth-class WsRF rating. Based on the results of the SHAP (SHapley Additive exPlanations) feature importance analysis [43], the X12 (Service name) and X13 (WSDL address) parameters were disregarded.

During the pre-processing stage of the evaluation of the QWS parameters, descriptive statistical values play a crucial role. Parameters in the dataset have their minimum, maximum, average, and standard deviations listed in Table 3. Incorporating the WsRF parameter into the model would lead it to over-fitting, while leaving it out has no effect on the quality of service. The service quality variables from the QWS dataset had an impact on the model estimation using the SHAP technique, and hence their relative importance in the estimation process dictated which parameters would be employed in the trials. Three of the nine parameters whose weights were calculated during estimation were removed from the final models in order to speed up their execution times. Proper hyperparameter tuning of the models is essential for optimal model performance. Random Search, Grid Search, Halvin Grid Search, Halvin Random Search, and Bayes Search were employed in the hyperparameter tuning phase [44].

Random Search serves to optimize the hyperparameters of the model by randomly selecting from predetermined ranges. It tries to achieve good results by trying different combinations of hyperparameters without following a specific pattern. It can be more inefficient than other methods, but can sometimes randomly reach good hyperparameter settings [45]. Grid Search creates a grid of possible hyperparameter values and allows systematic testing of all combinations. This method evaluates the performance of the model for each combination of

Table 2 QWS dataset parameters and descriptions

ID	Attribute name	Description	Unit
X1	Response *Time*	Time taken to send a request and receive a response	ms
X2	Availability	Number of successful invocations/total invocations	%
X3	Throughput	Total number of invocations for a given period of time	Invokes/s
X4	Successability	Number of response/number of request messages	%
X5	Reliability	Ratio of the number of error messages to total messages	%
X6	Compliance	The extent to which a WSDL document follows WSDL documentation	%
X7	Best *Practices*	The extent to which a web service follows	%
X8	Latency	Time taken for the server to process a given request	ms
X9	Documentation	Measure a documentation (i.e. description tags) in WSDL	%
X10	WsRF	Web service relevance function: a rank for web service quality	%
X11	Service classification	Levels representing service offering qualities (1 through 4)	Classifier
X12	Service name	Name of the web services	None
X13	WSDL address	Location of the web service definition language (WSDL) file on web	None

Table 3 Descriptive statistics of QWS parameters

Parameter	Mean	Std	Min	Max
Response_Time	384.01	564.55	142.50	4989.67
Availability	81.16	18.70	7.00	100.00
Throughput	9.03	7.73	0.10	43.10
Successability	83.90	19.90	8.00	100.00
Reliability	69.78	8.58	33.00	89.00
Compliance	88.43	10.02	33.00	100.00
Best_Practices	79.30	7.81	50.00	95.00
Latency	54.69	191.78	0.25	4140.35
Documentation	31.33	31.51	1.00	97.00
WsRF	0.72	0.08	0.21	0.94
Class	2.25	0.83	1.00	4.00

hyperparameters [46]. Halving Grid Search is an optimization technique that combines Grid Search and Random Search methods. It begins by evaluating a subset of randomly selected hyperparameter combinations. It then proceeds to select and evaluate the most promising combinations. This iterative process continues until the best performing combination is found [47]. Halving Random Search selects subsets of hyperparameter combinations using random sampling. It then proceeds iteratively to narrow the search space by focusing on the most promising combinations [48]. Bayesian Search uses probabilistic models to predict the performance of different combinations of hyperparameters. Finds optimal hyperparameters by minimizing the number of actual model evaluations [49]. The main purpose of these methods is to tune hyperparameters effectively and find the best configuration for a particular model.

For the purposes of this analysis, we split the QWS dataset into a training set of 70% and a testing set of 30%. Models were first developed using the training data, and then checked against the test data for accuracy. The following are characteristics of the experimental testbed employed in this study: Windows 10 Professional (64 Bit) OS, 16 GB DDR4 3200 MHz RAM, NVIDIA GeForce RTX 3050 graphics card, and an 11th-generation Intel Core i7-11800H processor (2.30 GHz). Jupyter Notebook was the best choice for the development environment. After finding the optimal parameters and hyperparameter values, the models were run one more time to acquire the final findings. The models' confusion matrix, classification report, and AUC data were compared.

4 Experimental results

In this section, we compare the proposed method to some of the most well-known classification algorithms used in service discovery research, including Naive Bayes, Decision Tree, KNN, SVM, and Random Forest. These algorithms are all Boosting algorithms, and we ran the experiments under the same conditions as Gradient Boosting, XGBoost, LightGBM, and CatBoost. The Naive Bayes algorithm yielded the lowest accuracy, 71.67%, while the CatBoost algorithm yielded the best accuracy at 90.42%. Experimental accuracy rates achieved with the methods utilized are shown in Fig. 2.

Other model comparison metrics include Recall, Precision, and F1-score values. F1-score is the average of precision and Recall. It shows the precision and robustness of the model. The SVM and CatBoost algorithms have the highest Recall values (89%) and the KNN algorithm has the lowest (66%) value. Naive Bayes has the lowest precision value (72%) and the Gradient Boosting method has the greatest precision value (92%). SVM and CatBoost methods have the greatest F1-score value, at 89%, while the KNN algorithm has the lowest, at 69%. Table 4 displays the values of accuracy, Recall, Precision, and F1-score derived from trials performed using classification methods. By comparing the acquired values, we find that the Boosting algorithms provide more reliable predictions than the alternatives and that the models perform admirably in terms of precision, responsiveness, and stability.

The receiver operating characteristic (ROC) curve was also utilized to facilitate an understanding of the findings. Since our models are multi-class, we used the one-to-one method to obtain AUC values rather than the generalized method used for binary classification models. One category is deemed "positive" while the others are deemed "negative" in this approach. With a 4-class data set, we calculated the AUC four times and averaged the results to get the model score. Mean AUC for CatBoost is 98.43, while it's just 91.12 for the Decision tree. The average AUC values

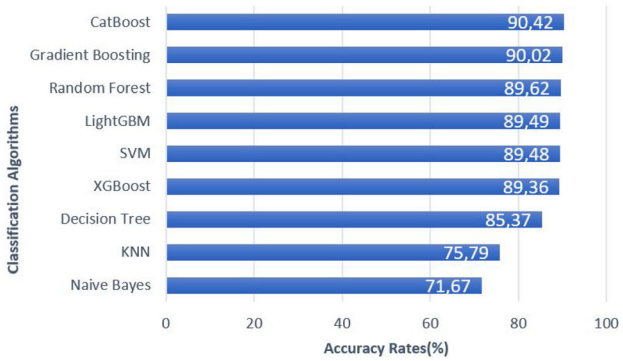

Fig. 2 Results obtained with classification algorithms

Table 4 Performance analysis of classification algorithms

Algorithms	Accuracy	Recall	Precision	F1-score
Naive Bayes	71.67	0.75	0.72	0.72
Random Forest	89.62	0.87	0.90	0.88
SVM	89.48	0.89	0.89	0.89
Decision Tree	85.37	0.84	0.86	0.85
KNN	75.79	0.66	0.75	0.69
Gradient Boosting	90.02	0.87	0.92	0.89
XGBoost	89.36	0.88	0.91	0.89
LightGBM	89.49	0.87	0.91	0.89
CatBoost	90.42	0.89	0.91	0.90

found during the studies are shown in Fig. 3. The average area under the curve for the various classification methods provides further evidence that the Boosting techniques are effective.

Table 4 and Fig. 3 show that CatBoost performs better compared to other algorithms, with an accuracy of 90.42% and an average AUC value of 98.43%. The CatBoost algorithm was executed using the first nine parameters present in the QWS dataset, and the influence of these parameters on the model predictions was determined as illustrated in Fig. 4. Within our ultimate model, the six parameters with the highest weights, namely Documentation, Successability, Throughput, Availability, Compliance, and Response Time, were employed in the experiments. The CatBoost method yielded the best results, and its AUC values were determined to be 0.9896 for the 1st Class, 0.9729 for the 2nd Class, 0.9792 for the 3rd Class, 0.9954 for the 4th Class, and 0.9843 on average. Analyzing the AUC values, we find that the developed model has high discrimination and accurate predictions. However, when looking at the ROC curve graphs for each

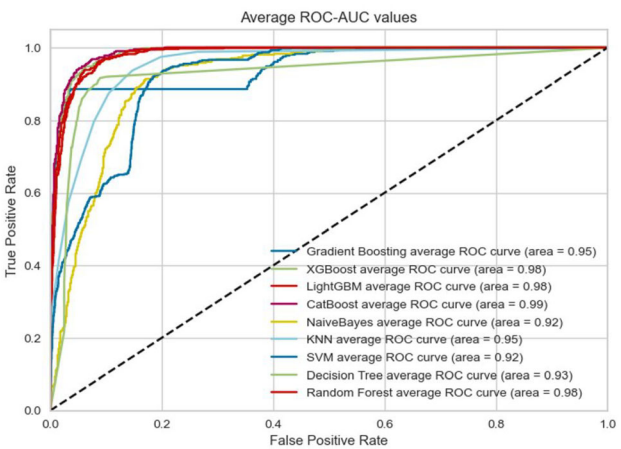

Fig. 3 ROC–AUC values of experiments

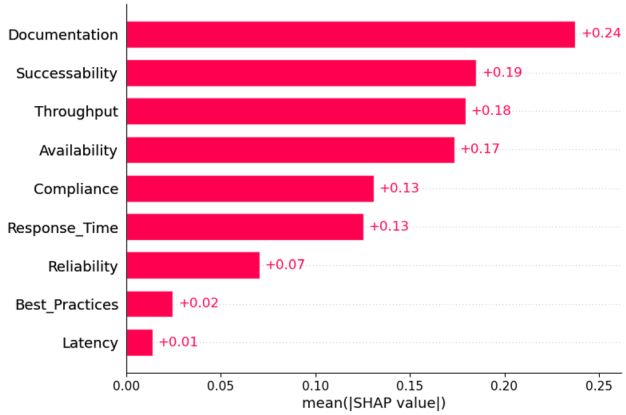

Fig. 4 CatBoost algorithm parameter weights

class in Fig. 5, it becomes clear that the 1st and 4th class curves of the model are superior to the others.

The confusion matrix in Fig. 6 provides the predictions provided by our model. An in-depth examination of the confusion matrix reveals that it predicts 96 of 114 first-class services in our model correctly, while wrongly classifying 18 others as second-rate. The model accurately predicted 345 out of 370 services in the second class but missed the mark on 11 out of 370 services in the first class and 14 out of 370 services in the third class. The model accurately predicted 180 of the 202 3rd-class services, mistakenly classifying 19 as 2nd-class and 3 as 4th-class. The confusion matrix reveals that the model had a high percentage of success (680/752) in its predictions, correctly classifying 59/66 services as 4th-class while incorrectly classifying 7 as 3rd-class.

5 Discussion

Initially, through evaluations among the selected algorithms from the literature, it was determined that the Random Forest algorithm exhibits superior performance. Additionally, it was observed that the Naive Bayes and KNN algorithms yielded the lowest success rates in our experiments. When investigating the reasons behind the comparative success of Random Forest and SVM algorithms, it becomes apparent that SVM excels due to its incorporation of correlated learning algorithms for analyzing data in classification problems. This attribute enables it to effectively capture relationships within data, leading to enhanced success in classification tasks. On the other hand, the Random Forest Algorithm, based on the principle of ensemble learning, possesses the ability to amalgamate results from diverse algorithms, thereby generating harmonious outcomes. This quality permits the amalgamation of strengths from various algorithms,

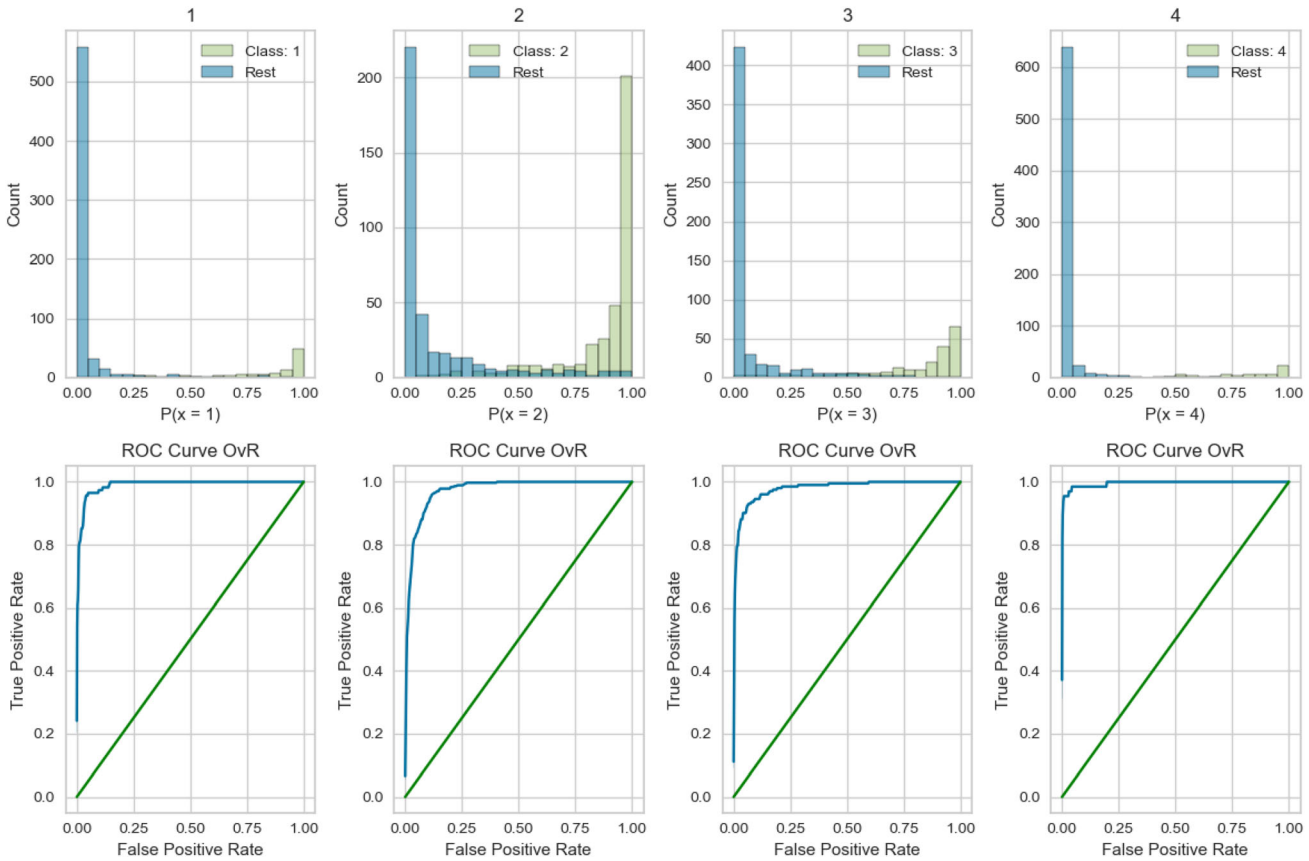

Fig. 5 CatBoost algorithm ROC curves and AUC values

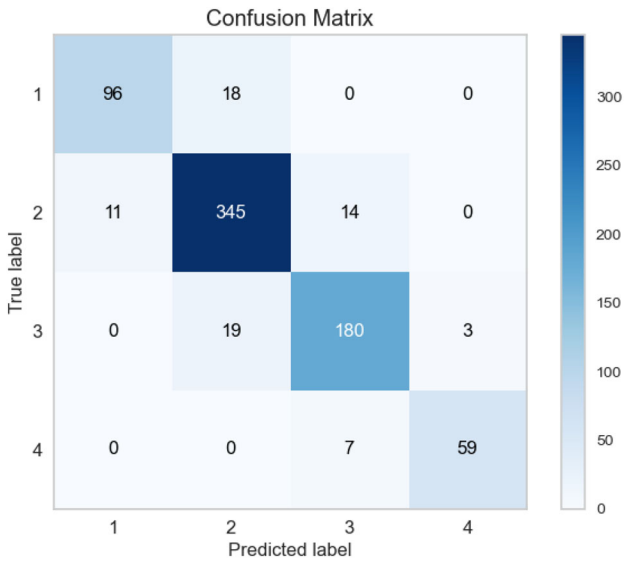

Fig. 6 CatBoost algorithm Confusion Matrix

subsequently yielding improved results. In this context, it can be postulated that algorithms operating based on ensemble principles, along with those that demonstrate superior performance on datasets with inherent relationships, have the potential to produce more advanced outcomes in the classification of microservices.

Experimental results show that boosting techniques perform well. Boosting algorithms are favoured because, like the Random Forest Algorithm, they are fundamentally ensemble learning methods and continue by correcting the faults produced in the prior learning. By comparing the acquired results, it is clear that the Boosting algorithms perform similarly to other algorithms and even outperform them. In a comparison of the nine algorithms utilized in the research, CatBoost comes out on top. Among Boosting algorithms, CatBoost is the newest algorithm in this field. Boosting algorithms are always being improved because of their usefulness in classification tasks. The AUC values also support this observation, providing the highest performance with the CatBoost algorithm.

When a comparative analysis is made between the amplification algorithms, it can be said that the use of amplification algorithms is generally user-friendly. However, it is observed that the process of finding the most suitable combination of hyperparameters to further improve the obtained results is laborious and complex. Especially in this comparison between the derived versions of the Gradient Boost algorithm, such as XGBoost and

LightGBM, it is seen that the Gradient Boost algorithm achieves better results, although there are slight differences. It can be emphasized that this situation may be closely related to both data properties and hyperparameter combinations. In contrast to XGBoost and LightGBM, we found the best results using CatBoost, which is an amalgamation of the two words—"Category" and "Boosting". CatBoost's most notable qualities are its adaptability to several types of data, including numerical, categorical, and textual data, as well as its GPU support, and other advanced capabilities. Its superior performance with categorical data is thanks to its novel coding approach. Data pre-processing is not only unnecessary but discouraged. Additionally, it captures a high prediction rate without creating very deep trees, hence avoiding over-learning, as a result of its symmetrical trees. This method also benefits from being able to acquire memory copies and learn with the GPU. These enhancements allow for smoother big-data operations.

In most cases, the results using Boosting algorithms were shown to have a high degree of accuracy. The high AUC values further demonstrated the uniqueness of these techniques. Boosting algorithms are advantageous since they are simple to implement. These algorithms feature in-built functionality to deal with missing data and do not necessitate any pre-processing of data. Boosting algorithms prioritize characteristics that improve prediction accuracy during training by decreasing the high deviations typical of machine learning models, and they do so successively, based on previous learning. Thus, massive datasets can be used more effectively and with fewer data properties. The algorithm has a few flaws. As each model attempts to remedy the mistakes of the one that came before it, it might cause distortions in the form of extreme values in the final product. Furthermore, because to its complex structure, utilizing Boosting techniques in real-time applications is challenging.

6 Conclusion

This study applied Boosting classification techniques to help end-users discover the best quality services to use in the microservice ecosystem. The experiments were run using the QWS dataset, which included traffic traces for 2507 operational real-world web service applications. The services have been classified according to the quality rating as follows: 1st Platinum (High Quality), 2nd Gold, 3rd Silver, and 4th Bronze (Low Quality). For predicting service quality, the six most influential parameters from the dataset have been selected, considering the weights on the model for each algorithm. Additionally, to determine the optimal results that each algorithm can produce, a combination of the best hyperparameter values for algorithms has been identified through five distinct hyperparameter tuning methods. Our web service discovery research has shown that boosting algorithms produce more accurate predictions compared to the most popular classification algorithms. Experimental results using Accuracy, Recall, Precision, F1-score, and Area Under the ROC Curve (AUC) parameters confirm that the boosting algorithms outperform the traditional machine learning methods. Among the nine classification algorithms used in the experiments, the most successful algorithm was the CatBoost classification algorithm with an accuracy of 90.42% and an AUC value of 98.43%. The lowest accuracy rate obtained from the experiments is the accuracy rate of 71.67% achieved with the Naive Bayes algorithm. CatBoost's capabilities such as running without the requirement for data preparation, learning with GPU, and working by making snapshots while running make it a promising candidate for further applications in future research. Researchers in this area will keep moving at a breakneck pace to identify the services that meet the necessary criteria in the fast-developing microservice ecosystem. In light of the fact that microservices are in a constant state of flux, it is crucial to use the most recent available datasets of the services to maximize the effectiveness of the research. Alternate classification algorithms and more recent datasets present opportunities to deepen and advance this research.

Funding Open access funding provided by the Qatar National Library (QNL).

Data Availability Data is publicly available from the following webpage: https://qwsdata.github.io.

Declarations

Conflict of interest The authors have no conflicts of interest to declare.

References

1. Akbulut, A., Perros, H.G.: Performance analysis of microservice design patterns. IEEE Internet Comput. **23**(6), 19–27 (2019)
2. Olewy, N.A.-H.H., Hadi, A.K.: Classifying quality of web services using machine learning classification and cross validation techniques. In: 2021 2nd Information Technology to Enhance E-learning and Other Application (IT-ELA), 2021, pp. 125–130. IEEE (2021)
3. Wang, H., Shi, Y., Zhou, X., Zhou, Q., Shao, S., Bouguettaya, A.: Web service classification using support vector machine. In: 2010 22nd IEEE International Conference on Tools with Artificial Intelligence, 2010, vol. 1, pp. 3–6. IEEE (2010)
4. Olewy, N.A.-H.H., Hadi, A.K.: Classifying quality of web services using machine learning classification and cross validation techniques. In: 2021 2nd Information Technology to Enhance E-learning and Other Application (IT-ELA), 2021, pp. 125–130. IEEE (2021)
5. Liu, J., Tian, Z., Liu, P., Jiang, J., Li, Z.: An approach of semantic web service classification based on Naive Bayes. In: 2016 IEEE International Conference on Services Computing (SCC), 2016, pp. 356–362. IEEE (2016)
6. Al-Masri, E., Mahmoud, Q.H.: Toward quality-driven web service discovery. IT Prof. **10**(3), 24–28 (2008)
7. Mohanty, R., Ravi, V., Patra, M.R.: Web-services classification using intelligent techniques. Expert Syst. Appl. **37**(7), 5484–5490 (2010)
8. Adarme, M., Jimeno, M.: QoS-based pattern recognition approach for web service discovery: Ar_wsds. Appl. Sci. **11**(17), 8092 (2021)
9. Madani, M.H., Youcef, A.: WSC2RCNN: a deep learning actions-based classifier for improved web service discovery. Comput. Sist. **26**(4), 1539–1548 (2022)
10. Huang, Z., Zhao, W.: A semantic matching approach addressing multidimensional representations for web service discovery. Expert Syst. Appl. **210**, 118468 (2022)
11. Swetha, N., Karpagam, G.: Lexicon ontology driven concept lattice framework for semantic web service discovery. In: 2022 6th International Conference on Computing Methodologies and Communication (ICCMC), 2022, pp. 1428–1435. IEEE (2022)
12. Sheng, H., Li, Z., Liu, J., Zhang, X.: Web service classification based on reinforcement learning and structured representation learning. In: 2021 International Conference on Artificial Intelligence and Blockchain Technology (AIBT), 2021, pp. 21–27. IEEE (2021)
13. El Allali, N., Fariss, M., Asaidi, H., Bellouki, M.: Towards semantic web services density clustering technique. In: International Conference on Digital Technologies and Applications, 2021, pp. 543–553. Springer (2021)
14. El Allali, N., Fariss, M., Asaidi, H., Bellouki, M.: Multinomial Naive Bayes categorization for semantic web services. In: 2021 International Conference on Digital Age and Technological Advances for Sustainable Development (ICDATA), 2021, pp. 74–79. IEEE (2021)
15. Li, S., Luo, H., Zhao, G., Tang, M., Liu, X.: Bi-directional Bayesian probabilistic model based hybrid grained semantic matchmaking for web service discovery. World Wide Web **25**(2), 445–470 (2022)
16. Zhang, X., Liu, J., Cao, B., Shi, M.: Web service classification based on information gain theory and bidirectional long short-term memory with attention mechanism. Concurr. Comput. Pract. Exp. **33**(13), 6202 (2021)
17. Kang, G., Xiao, Y., Liu, J., Cao, Y., Cao, B., Zhang, X., Ding, L.: Tatt-BiLSTM: Web service classification with topical attention-based BiLSTM. Concurr. Comput. Pract. Exp. **33**(16), 6287 (2021)
18. Huang, H., Cao, B., Liu, S., Zhou, D., Tang, M., Xiao, F.: A web service classification method based on graph neural network knowledge distillation. In: 2022 IEEE Smartworld, Ubiquitous Intelligence and Computing, Scalable Computing and Communications, Digital Twin, Privacy Computing, Metaverse, Autonomous and Trusted Vehicles (SmartWorld/UIC/ScalCom/DigitalTwin/PriComp/Meta), 2022, pp. 1710–1715. IEEE (2022)
19. Li, B., Li, Z., Yang, Y.: Residual attention graph convolutional network for web services classification. Neurocomputing **440**, 45–57 (2021)
20. Zhang, X., Liu, J., Shi, M., Cao, B.: Word embedding-based web service representations for classification and clustering. In: 2021 IEEE International Conference on Services Computing (SCC), 2021, pp. 34–43. IEEE (2021)
21. Agarwal, N., Sikka, G., Awasthi, L.K.: Web service clustering technique based on contextual word embedding for service representation. In: 2021 International Conference on Technological Advancements and Innovations (ICTAI), 2021, pp. 617–621. IEEE (2021)
22. Zhou, J., Jiang, B., Yang, J., Yang, J., Li, H., Wang, N., Wang, J.: Service discovery method based on knowledge graph and Word2Vec. Electronics **11**(16), 2500 (2022)
23. Sun, Q., Han, J., Ma, D.: A framework for service semantic description based on knowledge graph. Electronics **10**(9), 1017 (2021)
24. Tang, B., Yan, M., Zhang, N., Xu, L., Zhang, X., Ren, H.: Co-attentive representation learning for web services classification. Expert Syst. Appl. **180**, 115070 (2021)
25. Shi, M., Zhuang, Y., Tang, Y., Lin, M., Zhu, X., Liu, J.: Web service network embedding based on link prediction and convolutional learning. IEEE Trans. Serv. Comput. **15**(6), 3620–3633 (2021)
26. Alshafaey, M.S., Saleh, A.I., Alrahamawy, M.F.: A new cloud-based classification methodology (CBCM) for efficient semantic web service discovery. Clust. Comput. **24**, 2269–2292 (2021)
27. Alkalbani, A.M., Hussain, W.: Cloud service discovery method: a framework for automatic derivation of cloud marketplace and cloud intelligence to assist consumers in finding cloud services. Int. J. Commun. Syst. **34**(8), 4780 (2021)
28. Ouchaou, L., Nacer, H., Charoy, F.: Semantic-based cognitive service discovery in multi-cloud environments. In: 2022 First International Conference on Computer Communications and Intelligent Systems (I3CIS), 2022, pp. 82–87. IEEE (2022)
29. Swamy, K.: Web service classification using multi-layer perceptron optimized with Tabu search. In: 2015 IEEE International Advance Computing Conference (IACC), 2015, pp. 290–294. IEEE (2015)
30. Mustafa, A.S., Kumaraswamy, Y.: Data mining algorithms for web-services classification. In: 2014 International Conference on Contemporary Computing and Informatics (IC3I), 2014, pp. 951–956. IEEE (2014)
31. Das, M.S., Govardhan, A., Lakshmi, D.V.: Classification of web services using data mining algorithms and improved learning model. TELKOMNIKA (Telecommun. Comput. Electron. Control) **17**(6), 3191–3202 (2019)
32. Chippa, M., Priyadarshini, A., Mohanty, R.: Application of machine learning techniques to classify web services. In: 2019 IEEE International Conference on Intelligent Techniques in Control, Optimization and Signal Processing (INCOS), 2019, pp. 1–7. IEEE (2019)
33. Al-Masri, E., Mahmoud, Q.H.: Discovering the best web service: a neural network-based solution. In: 2009 IEEE International Conference on Systems, Man and Cybernetics, 2009, pp. 4250–4255. IEEE (2009)

34. Chandra, M., Niyogi, R.: Web service selection using modified artificial bee colony algorithm. IEEE Access **7**, 88673–88684 (2019)
35. Mhlanga, S.T., Chiyangwa, T.B., Lall, M., Ojo, S.: A hybrid-based architecture for web service selection. In: 2019 IEEE International Conference on Engineering, Technology and Education (TALE), 2019, pp. 1–8. IEEE (2019)
36. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., Liu, T.-Y.: LightGBM: a highly efficient gradient boosting decision tree. In: Advances in Neural Information Processing Systems, vol. 30 (2017)
37. Natekin, A., Knoll, A.: Gradient boosting machines, a tutorial. Front. Neurorobot. **7**, 21 (2013)
38. Mitchell, R., Frank, E.: Accelerating the XGBoost algorithm using GPU computing. PeerJ Comput. Sci. **3**, 127 (2017)
39. Namburu, A., Selvaraj, P., Varsha, M.: Product pricing solutions using hybrid machine learning algorithm. Innov. Syst. Softw. Eng. (2022). https://doi.org/10.1007/s11334-022-00465-3
40. Parameter tuning. https://catboost.ai/en/docs/concepts/parameter-tuning. Accessed 27 Jan 2023
41. About QWS dataset (2019). https://qwsdata.github.io/. Accessed 27 Jan 2023
42. Al-Masri, E., Mahmoud, Q.H.: QoS-based discovery and ranking of web services. In: 2007 16th International Conference on Computer Communications and Networks, 2007, pp. 529–534. IEEE (2007)
43. Van den Broeck, G., Lykov, A., Schleich, M., Suciu, D.: On the tractability of SHAP explanations. J. Artif. Intell. Res. **74**, 851–886 (2022)
44. Feurer, M., Hutter, F.: Hyperparameter optimization. In: Automated Machine Learning: Methods, Systems, Challenges, pp. 3–33. Springer, Cham (2019)
45. Zabinsky, Z.B., et al.: Random Search Algorithms. University of Washington, USA, Department of Industrial and Systems Engineering (2009)
46. Liashchynskyi, P., Liashchynskyi, P.: Grid search, random search, genetic algorithm: a big comparison for NAS. arXiv preprint (2019). arXiv:1912.06059
47. Nagaraj, B., Malagi, K.B.: Boosting the accuracy of optimisation chatbot by random forest with halving grid search hyperparameter tuning. ICTACT J. Soft Comput. **13**(3), 3007–3013 (2023)
48. Bakır, H., Çayır, A.N., Navruz, T.S.: A comprehensive experimental study for analyzing the effects of data augmentation techniques on voice classification. Multimed. Tools Appl. (2023). https://doi.org/10.1007/s11042-023-16200-4
49. Alibrahim, H., Ludwig, S.A.: Hyperparameter optimization: comparing genetic algorithm against grid search and Bayesian optimization. In: 2021 IEEE Congress on Evolutionary Computation (CEC), 2021, pp. 1551–1559. IEEE (2021)

Ahmet Vedat Tokmak completed his undergraduate education in Computer Systems Education at Gazi University and Computer Engineering at Namık Kemal University. He earned his master's degree in Computer Engineering from Istanbul Kültür University. He currently serves as an Information Technology teacher within the Ministry of National Education, where he is dedicated to guiding students in the fields of computer science and technology.

Akhan Akbulut received the B.S. and M.S. degrees in computer engineering from Istanbul Kültür University (IKU), Turkey, in 2001 and 2008, respectively, and the Ph.D. degree in computer engineering from Istanbul University, Turkey, in 2013. From 2004 to 2013, he was a Research Assistant with the Department of Computer Engineering, IKU, where he served as an Assistant Professor from 2013 to 2017. From 2017 to 2019, he worked as a Postdoctoral Researcher with the Department of Computer Science, North Carolina State University, NC, USA. In 2019, he rejoined the Department of Computer Engineering, IKU, and was promoted to the position of Associate Professor. Currently, he serves as the Chair of the Department of Computer Engineering, IKU. His current research interests include the design and performance optimization of software-intensive systems, machine learning applications, Internet architectures, and advancing participation in cloud computing research.

Cagatay Catal is a Full Professor at Qatar University, Department of Computer Science & Engineering. He obtained his BSc and MSc degrees in Computer Engineering from Istanbul Technical University in 2002 and 2004, respectively. He later pursued his Ph.D. in Computer Engineering at Yildiz Technical University in Istanbul, completing it in 2008. From 2020 to 2021, he held the position of Full Professor in the Department of Computer Engineering at Bahcesehir University in Istanbul. Prior to that, he spent two years as a full-time faculty member at Wageningen University & Research (WUR) in the Netherlands. Before his time at WUR, he worked for six years in the Department of Computer Engineering at Istanbul Kultur University, where he served as an Associate Professor and Head of the Department. In April 2014, he was awarded the title of Associate Professor by the Inter-University Council of Turkey. Before entering the academic field, he had 8 years of experience at the

Scientific and Technological Research Council of Turkey (TUBITAK), Information Technologies Institute, where he held the positions of Senior Researcher, Project Manager, and Senior Software Engineer. His research interests encompass various areas, including Artificial Intelligence, Deep Learning, Machine Learning, Natural Language Processing, Software Engineering, Software Testing, Software Architecture, and Precision Agriculture.