Automated Software Engineering (2023) 30:21
https://doi.org/10.1007/s10515-023-00388-8

Machine learning based predictive modeling to effectively implement DevOps practices in software organizations

Ankur Kumar[1] **· Mohammad Nadeem**[1] **· Mohammad Shameem**[2]

Received: 12 October 2022 / Accepted: 18 June 2023 / Published online: 12 July 2023

Abstract

Development and Operations (DevOps) is a relatively recent phenomenon that can be defined as a multidisciplinary effort to improve and accelerate the delivery of business values in terms of IT solutions. Many software organizations are heading towards DevOps to leverage its benefits in terms of improved development speed, stability, collaboration, and communication. DevOps practices are essential to effectively implement in software organizations, but little attention has been given in the literature to how these practices can be managed. Our study aims to propose and develop a framework for effectively managing DevOps practices. We have conducted an empirical study using the publicly available HELENA2 dataset to identify the best practices for effectively implementing DevOps. Furthermore, we have used the prediction algorithms such as Support Vector Machine (SVM), Artificial Neural Network (ANN) and Random Forest (RF) to develop a prediction model for DevOps implementation. The findings of this study show that "Continuous deployment", "Coding standards", "Continuous integration", and "Daily Standup" "are the most significant practicesduring the life cycle of projects for effectively managing the DevOps practices. The contribution of this study is not only limited to investigating the best DevOps practices but also provides a prediction of DevOps project success and prioritization of best practices. It can assist software organizations in getting the best possible practices to focus on based on the nature of their projects.

Keywords DevOps · Prediction model · Support vector machine · Artificial neural network · Random forest

✉ Mohammad Shameem
shameem.ism@gmail.com

[1] Department of Computer Science, Aligarh Muslim University, Aligarh, India

[2] Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Andhra Pradesh, India

1 Introduction

Over the last ten years, innovation and agility have become essential for success in a highly competitive business environment, where concepts, technology, tools and timelines are constantly changing (Brand et al. 2021; Dehgani and Navimipour 2019). DevOps has emerged as a methodology for handling and adapting to the inevitable changes in requirements brought about by business demands throughout the software lifecycle (Stahl et al. 2017). DevOps is a set of practices that combines software development (Dev) and IT operations (Ops) (Morales et al. 2018). It follows a short development iteration and provides continuous delivery with high software quality (Shahin et al. 2017). The shorter lifecycle and fewer resources required for software development indeed resulted in accelerating software development (Fitzgerald and Stol 2017). The practices and principles associated with DevOps evolved with time to get the claimed benefits (Azad 2022). DevOps has several benefits over the traditional approaches, i.e., bridging the existing gap between development and IT operations teams, effective resource utilization, low development cost, waste management, fast project delivery, and minimizing human efforts (Faustino et al. 2022; Almeida et al. 2022).

Software organizations often turn towards the latest/modern software development practices for the likelihood of project success in terms of customer satisfactions, early project delivery and quality services and product (Lin et al. 2022). Implementing a new approach before an organization has determined which components may be helpful and which may be harmful carries a significant risk of losing time and money. DevOps methodology, which is ever-improving to beat the competition and deliver transformational change in core operations, is replacing traditional and agile methods as the preferred choice among leaders of software organizations (Dörnenburg 2018). Over the last decade, DevOps concept has emerged as one of the most popular processes for developing a quality software product. However, still, there is a lack of literature and quantitative studies that point to the combination of best practices and methodological traits leading to success (Erich et al. 2017). DevOps practitioners may not be aware of whether their adopted strategies, practices, skills, culture, and tools are suitable to achieve the benefits claimed by DevOps (Mumbarkar and Prasad 2022; Lazuardi et al. 2021). If the effort to implement DevOps fails, it leads to a wastage of human as well as financial resources and lowers overall morale and productivity (Senapathi et al. 2018). Therefore, to implement DevOps effectively, it becomes necessary to understand where an organization currently performs by measuring performance and creating a baseline (Narang and Mittal 2022). As practitioners of DevOps, many software organizations require greater expertise or tools to comprehensively measure their adopted practices and capabilities and therefore seek external assessment options (Badshah et al. 2020).

Currently, Artificial Intelligence and Machine Learning applications are expanding their wings in the IT business, allowing organizations to boost business values (Amershi et al. 2019).The use of machine learning-based approaches in DevOps is revolutionizing both DevOps and software development

methodologies (Bruneliere et al. 2022). Gartner reported 40% of DevOps teams would use machine learning-based techniques to evaluate their application and infrastructure by 2023 (Sabharwal and Bhardwaj 2022). DevOps teams can save a significant amount of time by using ML-driven requirements management solutions, allowing them to focus more on producing software products on time (Karamitsos et al. 2020). Data scientists and ML engineering teams are thus studying DevOps techniques to create ML-based predictive models that will boost the entire organization's value. Nogueira et al. (2018) highlighted ongoing research and addressed the potential application of machine learning technologies to improve the quality of the processes and outputs inside the DevOps pipeline. Angara et al. (2018) developed a predictability model for automated testing to predict success using machine learning algorithms. In his work, Wanget al. (2020) discussed the research and design of intelligent DevOps platforms based on machine learning. Utilizing machine learning techniques to conduct analysis and develop an intelligent DevOps platform will increase the efficiency of DevOps developers.

Various studies have been conducted related to the DevOps implementation in software organizations but there is still a research gap related to the roadmap to guide and recommend the DevOps practitioners to implement DevOps applications in their organizations (Amaro et al. 2022; Callanan and Spillane 2016; Pianini and Neri 2021; Macarthy and Bass 2020) effectively. DevOps practitioners still lack precise and clear guidelines to follow while implementing DevOps applications in their organizations (Leite et al. 2019; Gall and Pigni 2022). In currently available studies, the models used to assess the efficacy of DevOps were based either on information obtained from DevOps professionals or on information produced through DevOps process analysis (Samarawickrama and Perera 2017; Lwakatare et al. 2019). The information gathered by the authors was insufficient to construct a maturity or effectiveness model for a comprehensive assessment of all DevOps-related activities. Furthermore, there are not many examples in the literature of using machine learning methods to build predictive models that can predict DevOps success based on the methodologies and practices employed (Subramanya et al. 2022). The results of Machine Learning based assessment models depend on data collection, preparation, pre-processing, model training, testing, and validation (Marijan et al. 2019). Software organizations using DevOps need a framework to quantify the contribution of the practices, capabilities and approaches adopted in achieving maximum DevOps effectiveness (Kirk 2022). Therefore, using evaluation models based on machine learning prediction to forecast the effectiveness of practices, capabilities, and approaches used can help practitioners review the capabilities they adopt (Castellanos et al. 2021).

The present study aimed to determine the effectiveness of software development practices in addressing the goals of DevOps adoption in software organizations through the use of machine learning prediction algorithms. Information collected for the HELENA2 dataset was used for this purpose (Klünder, et al. 2020). The HELENA2 dataset itself is comprehensive because it was created by a team of 75 researchers to gather input from respondents working in software sectors in 55 different countries. The size and quality of the dataset allow machine learning

prediction techniques to be used. SVM, ANN, and Random Forest are the three types of machine learning models developed and validated to predict the effectiveness score based on 20 input factors (the use of DevOps + 19 software practices). The part of the dataset that deals solely with the DevOps approach has been used in this study. There are 296 responses left after clearing the data to determine DevOps-oriented data. We have discussed the preprocessing of the dataset in the research methodology part of the paper. In addition, we have implemented machine learning prediction algorithms on the portion of the dataset determined for DevOps. We also conducted a sensitivity analysis to determine the critical software practices that can be adopted in software organizations to implement DevOps. To fulfill the objective of our study, we framed the following research questions:

RQ1. How can the effectiveness of the DevOps implementation be measured?

RQ2. How can a machine learning-based predictive model be developed to evaluate the effectiveness of DevOps?

RQ3. How can the performance of the prediction model be measured?

The remainder of the paper is structured as follows: Sect. 2 discusses the motivation for this study. In Sect. 3, the research methodology is discussed. Section 4 presents the results of the study, and discussion of results in Sect. 5. Limitations of the study are listed in Sect. 6. Section 7 presents the implications of the study, and the final Sect. 8 presents the conclusion.

2 Motivation

The idea of DevOps was first introduced in 2008 by Andrew Clay and Patrick Debois (Hemon et al. 2020) The purpose of introducing DevOps was to overlap the shortcomings of Agile (Shameem et al. 2023). Agile development works well and efficiently with the development team but does not address the issues the operations team faces, which lengthens development time and costs (Khan et al. 2021). In Agile, most of the pressure falls on developers and there is less space for operators to contribute (Shameem et al. 2020). Due to the lack of commitment among software development stakeholders, most projects get delayed (Rodríguez et al. 2019).

Many studies have focused on different aspects of defining DevOps culture and successfully implementing it in software development. Lwakatare (Lwakatare et al. 2019) conducted a literature survey and empirical study using interview techniques with DevOps practitioners to identify the elements underpinning the phenomenon of DevOps. The findings of their study reported that collaboration among the team members, measurement (to measure DevOps pipeline performance), automation, and monitoring were the four metrics identified. Smeds (2015) defined DevOps as the attributes, capabilities, culture, and technology that brings the Dev and Ops teams closer to work collaboratively. Erich (2017b) identified seven areas such as sharing, automation, the culture of collaboration, measurement, services, quality assurance, and governancerelated to DevOps. The authors explored these factors by conducting case studies in the six organizations for the implementation of DevOps in practice. Luz (2019) formulated a theory on DevOps adoption. They found that DevOps activities have a correlation between

the factors i.e., collaborative culture, agility, automation, continuous evaluation, quality assurance, flexibility, sharing, and transparency.

DevOps culture attracts software organizations to deliver its benefits, but at the same time, the unavailability of a clear way to implement it effectively creates a sense of hesitation in software organizations (Benni et al. 2019). Implementation of DevOps brings with it both benefits and challenges.Effective implementation is not an easy and straightforward task for organizations due to the challenges associated with it. The authors began to focus on identifying the benefits and challenges of adopting DevOps practices. Arif and Shamim (Khan and Shameem 2020) identified key challenges that could affect the implementation and adoption of DevOps practices within software organizations. In their study, Khan (2022) and others identified the cultural challenges faced by organizations when they adopted DevOps as well as defined the functioning of DevOps in a software organization.The challenges, benefits, and practices of DevOps outlined in many research studies can be used to implement DevOps effectively (Khan et al. 2022).

Software organizations need a way that can bypass challenges and transform their processes, technologies, or skills to reach the highest level in the DevOps journey (Anandya et al. 2021) As a practitioner of DevOps, software organizations are eager to know how effective the adopted skills, tools, and techniques are in achieving the elite level of benefits of DevOps (Shameem 2022) Therefore, to achieve the benefits promised by DevOps, the determination of the effectiveness of the practices, skills, and tools employed, can help the practitioners to review their adopted skills and tools.

The issues that hinder the effective implementation of DevOps can be easily handled by striking a balance between the benefits and the challenges (Mirina et al. 2019). There is a need for a model that can measure the effectiveness of software development practices by evaluating how much the adopted practices benefit or present a challenge. Our study is an initiative toward proposing a model that assesses the effectiveness of DevOps practice. The proposed model is a data-driven machine learning prediction model for assessing the effectiveness of DevOps practices.

3 Research methodology

The research method used in the present study was implemented in three phases: Research objectives, Data preprocessing, and developing prediction model as shown in Fig. 1. In Phase 1, the related work, research objectives are discussed. The Phase 2 involves the data preprocessing which is needed to be supplied in the prediction algorithms.The preparation of the data includes the determination of DevOps-related data from the HELENA2 dataset. Subsequently, in the third phase, data obtained from the dataset related to DevOps was supplied to various machine learning prediction algorithms to train the model and develop a prediction model to measure the effectiveness of related software practices in implementation across software organizations.

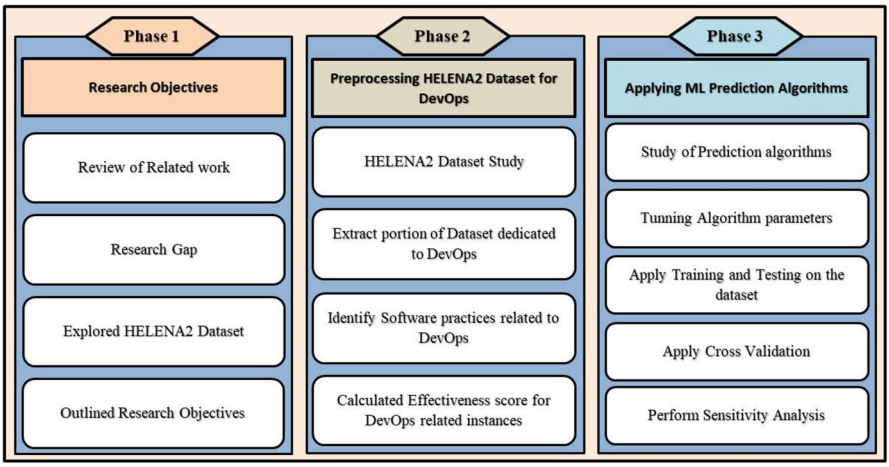


Fig. 1 Phases of research methodology

3.1 Phase-1: Defining research objectives

3.1.1 Review of related work

In the literature, there are some research articles, especially with regard to the context of our study objectives. Bijwe and Shankar (Bijwe and Shankar 2022) discussed the DevOps practices implementation in IoT projects. The authors of the study highlighted the impact of DevOps on the performance metrics of IoT applications. Catherine Crowley et al. (2018) have developed the DevOps Effectiveness Assessment (DEA) model to illustrate a comprehensive analysis of DevOps maturity within software development organizations. The model was developed after rigorous research and collaboration of experts and academics from leading organizations. The DEA is built with the help of nine DevOps principles and aims to guide DevOps professionals to transition to DevOps. Heine (2022) introduced multivariate regression models with the ability to predict the effectiveness of DevOps implementations in combination with other software development frameworks and practices. His research aimed to feed IT professionals with quantitative data that will ultimately help in making decisions about the implementation of DevOps, other approaches that complement DevOps, and frequently used practices for effective project management. Ivanova and Ivanov (2018) have used the data collected from DevOps processes to improve the development capabilities of enterprises. The metrics and KPIs for measuring DevOps effectiveness are also described in this paper. The authors show the application of machine learning algorithms to classify code change requests into cost classifications as an example of KPI analytics. In addition, the paper also explored the switching role of a DevOps engineer with respect to data analytics and highlighted the skills and knowledge needed to be successful in the era of big data. Forsgrenetal. (2018) conducted a hierarchical cluster analysis using a sample of 7522 DevOps professionals, examining how stability and throughput measurements work together

and creating three different software delivery performance profiles of development settings. Marrero and Astudillo (2021) described the DevOps-RAF Assessment Framework for evaluating DevOps practices, culture, and tools. The paper's authors also identified capacity gaps and proposed roadmaps for improvement. They have validated the model with the help of a survey of several DevOps industry experts. Marijan et al. (2018) proposed a technology that has integrated risk and defect-oriented test selection and optimized priority to improve the effectiveness of continuous integration testing and thus reduce DevOps' long-term cycle times.

DevOps literature exploration has made us aware that DevOps is transforming work culture, increasing collaboration and communication, and automating processes at multiple stages of the software development lifecycle (Gheorghe-Pop et al. 2020). The benefits, challenges, and practices of DevOps elaborated in various research articles can be used to implement DevOps effectively. While exploring the studies focused on quantifying DevOps effectiveness using machine learning techniques, we found the above-mentioned studies during the literature study. The reported experience of such DevOps effectiveness models is very limited. The data used in existing studies that refer to the effectiveness of DevOps is based either on data gathered from DevOps experts or on data derived from analysis of DevOps processes. The quality and quantity of data employed in the development of the DevOps effectiveness assessment framework were not sufficient for an overall evaluation and assessment of DevOps effectiveness in software organizations.

3.2 Research objectives

Inspired by the research gap, it has been realized a need to assess or evaluate the effectiveness of software development practices in addressing the goals of DevOps adoption in software organizations (Akbar et al. 2020) There is a scarcity of studies employing machine learning prediction algorithms to predict the effectiveness of DevOps based on DevOps-related data (Gupta et al. 2017) We had a HELENA2 dataset which contains data about DevOps and can be utilized to input into machine learning prediction algorithms to predict the effectiveness of DevOps. This study is done with the following objectives:

- To identify software practices that best serve the purpose of DevOps adoption within software organizations.
- To determine the effectiveness of identified DevOps practices in implementing DevOps within a software organization.
- To propose a model for predicting the effectiveness of DevOps in software development organizations.

3.2.1 HELENA2 dataset identification

We discovered datasets compiled by the Helena group on the hybrid software development technique utilized in the software development enterprise when exploring the literature (Kuhrmann et al. xxxx) The dataset was named as HELENA2 Dataset.

We found this data to be suitable for meeting our research objectives. This dataset contains a decent amount of data collected from respondents from the software industry located around the world.

3.3 Preprocessing dataset for machine learning algorithms

Machine learning prediction algorithms require a set of inputs called features for the input purpose with corresponding outputs (Lwakatare et al. 2020). This section of the paper discusses the pre-processing of the HELENA2 dataset to determine the dataset portion that is entirely DevOps-related and according to the requirements of machine learning algorithms.

3.3.1 HELENA2 dataset description

HELENA is a large-scale international survey that has collected a good amount of data in response to the general and hybrid use of various software approaches for software development.HELENA aims to determine the current state of software practices used in different software development approaches. A team of 75 researchers was employed to collect data from approximately 1,500 project/product teams around the world to develop the HELENA2 dataset (Kuhrmann et al. 2018). The dataset presents a worldwide report on current practice as it was done in an international environment collecting data from 55 countries. Dataset was stored in an Excel spreadsheet containing responses from 1467 participants about 211 data points. This dataset is publicly accessible, which inspires further research. The dataset specified 24 software development approaches as specified in section PU09 to seek feedback from participants belonging to software development organizations. It has considered all the development methodologies or frameworks developed so far. The dataset also took feedback from participants about 36 development practices to associate with each of the 24 software practices outlined in the PU10 section of the dataset (Khan et al. 2022).In addition, the dataset also includes the 18 targets displayed in section PU12 that seek feedback from participants to assign degrees to each of the 18 targets. Apart from DevOps, there are other development frameworks in this dataset as well. Hence this dataset needs to be pre-processed to determine the dataset that is completely DevOps related. The preprocessing step is depicted in Fig. 2 and is explained in detail in the later Sects. 3.2.2, 3.2.3, and 3.2.4.

3.3.2 DevOps oriented dataset extraction

Firstly, we have analyzed the entire dataset to understand the meaning of the stored values of different data points in an Excel spreadsheet. The dataset contains a rich collection of data for DevOps as well as 24 software development frameworks. The data was classified to separate the DevOps-related information from that larger dataset. Then, we isolate the portion of the dataset solely related to DevOps by examining each question contained in the data points.DevOps-related datasets were identified with the help of answers to questions about the use of

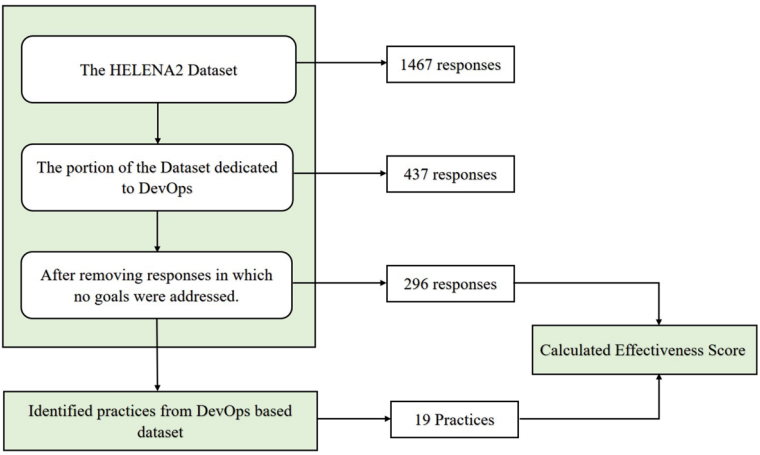


Fig. 2 Preprocessing of HELENA2 Dataset for DevOps

DevOps methodology. The feature depicting the 'Use of DevOps' was mentioned in subsection PU09_03 of the Process Usage (PU) section.The options for using DevOps range from 1 to 7 to be provided for participants to register their feedback, meaning: 1 = don't know the framework, 2 = don't know whether we use it or not, 3 = we never use it, 4 = we use it rarely, 5 = we use it rarely, 6 = we use it often, 7 = we use it always. We restrict ourselves to the part of data with values 4,5,6,7 from the whole dataset to extract data related to DevOps.This reveals that we have considered instances of respondents who said about the 'Use of DevOps', such as 'we rarely use DevOps', 'we rarely use DevOps', 'we often use DevOps', and 'we always use DevOps'.These options were considered because they reflect the spirit of DevOps usage.We found 437 out of 1467 instances as a part of the dataset identified for DevOps.These 437 responses were classified into four categories namely 'Rarely Used DevOps', 'Sometimes Used DevOps', 'Frequently Used DevOps', 'Always Used DevOps', as shown in Table 1.The question regarding the use of DevOps is specified in subsection PU09_03 of the Process Usage (PU) section, which was again used as a column when preparing the DevOps-related dataset.

Table 1 Number of responses for different classes of DevOps

Use of DevOps	No. of responses	Percentage (%)
4: Rarely used DevOps	85	19
5: Sometimes used DevOps	138	32
6: Often used DevOps	135	31
7: Always used DevOps	79	18
Total	437	100

3.3.3 Identification of DevOps related software practices

After identifying the portion of responses devoted to DevOps, we have attempted to extract the widely used software development practices required for implementing DevOps. To identify software development practices related to DevOps, the frequency analysis method was adopted.The dataset took feedback from participants for 36 development practices, which are specified in the PU10 section of the dataset.A wide range of options was available for each software development practicelike 1 = don't know the framework, 2 = don't know if we use it, 3 = we never use it, 4 = we rarely use it use, 5 = we use it occasionally, 6 = we use it often, 7 = we use it always.We have decided to record the frequency of responses of the participants who chose between 4 to7, i.e., rated the software practice as rarely used, sometimes used, often used, and always used. The frequency under the columns of rarely used, sometimes used, often used, and always used was reviewed and placed the software practice in the maximum frequency category.We have received 2 software practices as rare practices, 7 software practices as sometimes used practices, 6 software practices as often used practices, and 13 software practices as always used practices. Of these, 6 frequently used and 13 always used practices are considered in the present study.A total of 19 practices related to DevOps have been identified as shown in Table 2. Table 2 summarizes the percentage of responses from participants who had selected between 1 to 7, out of 437 DevOps related responses. The definitions of options 1 to 7 are provided in Table 2.These 19 software development practices will be used as a set of features to prepare input data for machine learning prediction algorithms. The columns containing these 19 software practices are compiled with the portion of the dataset that was determined for DevOps. The stored data corresponding to these 19 practices will later be used as input features in machine learning prediction algorithms. A few majors of these practices are discussed next.

(a) Architecture specifications: Software architecture is an essential artifact for both developers and operators. The physical separation of multiple environments (e.g., development environment, test environment, production) makes it difficult for enterprises to adopt the standard DevOps architecture directly (Liu et al. 2021). The foundation of DevOps practices is based on the step-by-step idea of software architecture. Recently introduced architectural styles such as microservices, containerization, and orchestration have emerged to address the needs of scalability, deployment, and continuous delivery.
(b) Automated unit testing: DevOps has emerged to strengthen testing practice with an emphasis on continuous and automated testing. Automated unit testing is a standard process that minimizes the number of defects in the software system which is suitable for achieving the purpose of DevOps (Rafi et al. 2022). These tests are performed to ensure that individual services or packages are working correctly and check whether the individual parts work together as they should.
(c) Backlog management: The Product Manager prioritizes the Product Backlog as one of the pre-development activities. The Development and Operations team pulls items from the Product Backlog on a priority basis and uses them for iteration/sprint planning. If an item needs preventive or corrective meas-

Table 2 Identified software practices related to DevOps along with percentage of usage

ID	Software practices	Do not know the practice (1) (%)	Do not know if we use it (2) (%)	Never used practice (3) (%)	Rarely used practice (4) (%)	Sometimes used practice (5) (%)	Often used practice (6) (%)	Always used practice (7) (%)
SP1	Architecture Specifications	5	4	8	14	21	30	18
SP2	Automated Unit Testing	1	2	4	12	19	29	32
SP3	Backlog Management	2	2	6	5	15	30	40
SP4	Burndown Charts	7	4	11	13	19	21	26
SP5	Code Review	0	0	2	10	20	27	41
SP6	Coding Standards	1	1	2	7	17	34	38
SP7	Collective Code Ownership	13	6	10	9	14	20	28
SP8	Continuous Deployment	2	3	12	15	20	25	23
SP9	Continuous Integration	0	2	6	9	16	29	36
SP10	Daily Stand-up	3	1	7	8	13	22	46
SP11	Definition of Done / Ready	8	5	6	9	14	26	32
SP12	Expert/Team-based Estimation	4	4	8	9	17	33	25
SP13	Formal Estimation	5	4	13	12	18	26	22
SP14	Iteration/Sprint Reviews	4	3	7	11	15	28	34
SP15	Limit Work InProgress	2	1	7	8	15	28	40
SP16	Refactoring	1	1	4	11	30	35	17
SP17	Release planning	0	2	5	8	18	34	34
SP18	Retrospectives	4	2	7	8	16	25	37
SP19	User Stories	3	3	7	10	19	29	30

ures, it must be sent to the Product Backlog along with the assessment reports (Çalikli et al. 2018). Otherwise, all the aggregated items in the Product Backlog are forwarded to go through Quality Gates on the DevOps Pipeline.
(d) Burn down charts: Burndown charts are used in flexible development frameworks such as DevOps, Agile, Kanban, and Scrum. Burndown charts graphically represent an estimate of how much work is left to be done versus time or the number of tasks completed versus incomplete (Chakraborty Bapi and Karthikeyan 2019). The project manager can view and understand the progress of the project through burndown charts so that adjustments can be made to the work progress.
(e) Code review: Code review is the activity of submitting source code changes for approval, comment, and improvement. Factors considered important in a code review include the reviewer's knowledge, the size of the code changes to be reviewed, and review environment support (Saidani et al. 2021). The use of automation in code review is encouraged to achieve the goals of DevOps.
(f) Coding standards: DevOps has established the fact that not only functionality can be coded, but infrastructure or environmentcan also be configured with code (Albuquerque and Cruz 2019). The shifting left concept is promoted by DevOps, where the operations team starts working on the code phase. The static code analyzer verifies performance with coding standards and continuously informs about the best practices to be used or codification according to norms of object-oriented programming language.
(g) Collective code ownership: DevOps allows collective ownership of code, meaning that the source code is always available to any member of the team, at any time, which is implemented through version control tools. Collective code ownership allows teams to do the necessary software development on their own independent systems. Teams must be aligned at the product level to use knowledge sharing and mastery processes to support collective code ownership (Rahman et al. 2019).
(h) Continuous deployment:. Continuous deployment and delivery are the ultimate objectives of implementing DevOps.The concepts and principles of DevOps help break down the silos within software delivery organizations to achieve this continuous deployment. Continuous Deployment is different from Continuous Delivery because it involves automated deployment in a production environment where the end-user will have accessibility to newly released features (Soni 2015). Thus, Continuous Deployment should be tasked with fully automating production deployment.
(i) Continuous integration: DevOps enables continuous integration throughout the software development lifecycle by supporting the artifact changes needed to create a continuous pipeline. Continuous Integration is the repetitive act of integrating the practices of development and testing throughout the process of software development. Continuous Integration minimizes risks such as late detection of defects, lack of deployable software, and low project visibility (Prado Lima and Vergilio 2020). In CI, code committed to a version control repository is often pushed to the CI server, and build scripts are generated to integrate new changes into the software.

(j) Daily standup: Agile practices such as daily stand-up meetings, sprint planning meetings, and Scrum of Scrums were also adopted in DevOps to enable communication and coordination across developer and operations teams. Daily stand-up meetings are helpful for managing task allocation dependencies and expertise dependencies (Hemon et al. 2020). In these meetings, each team member tells the other members what they have done since the last daily stand-up meeting.

3.3.4 Effectiveness score calculation

The dataset also took feedback on the extent to which the respondent addressed the 18 goals outlined in the PU13 section of the dataset (please refer Table 5b). Each participant has given the option of assigning degrees to 18 goals on a scale of 1 to 10. Some of the participants have not assigned degree to none of the goals.We have removed those instances in which no goal was marked by the respondents. The 296 instances left after deleting the rows addressed zero goals.We have calculated effectiveness scores for 296 responses that were identified as completely DevOps related. For the calculation of effectiveness scores, the degree of addressing the 18 goals present in section PU13 of the dataset has been used.Each goal could be addressed in a range from 1 to 10 where 1 meant the goal was 'not achieved at all' while 10 meant 'fully achieved'. The data points available in section PU12 were used to store the number of goals that were later used in the calculation of effectiveness scores. The variable 'Number of goals addressed' will have a value between 0 and 18. For the number of goals addressed, if the assigned value is 0, it means that none of the goals were addressed, and 18 means that all goals were addressed. Another new variable 'Sum of the degrees of the goals addressed' is generated by adding up all the degrees specified by the respondents for each goal.The effectiveness score was then calculated by dividing the sum of the degrees to which the goals were addressed by the total number of goals addressed multiplied by 10 (Heine xxxx).

$$\text{Effectiveness Score} = \frac{\text{Sum of degrees of all Goal addressed}}{(10*\text{Total number of Goal addressed})} \tag{1}$$

The raw dataset contained 1467 records, but Effectiveness scores were calculated for DevOps related 296 records. The effectiveness score value for each record range from 0 to 1 or 0% to 100% effective. A new column has been added to the dataset to represent the effectiveness score corresponding to each of the 296 responses (Table 3). This calculated effectiveness score corresponding to each row will be used in the form output feature in the application of machine learning prediction algorithms.Since the goals were used to calculate effectiveness scores and goals are achieved through adopting software practices, this means that software development practices have a direct impact on effectiveness scores. The effectiveness score indicating the degree of achieving goals is dependent on the adoption of software development practices.

The final version of the dataset that underwent the machine learning prediction algorithm includes 296 rows (participants' responses) and 21 columns (19 for software practices, 1 regarding use of DevOps, and 1 for effectiveness scores

Table 3 Number of responses for different classes DevOps with 296 cases

Use of DevOps	No. of respondents	Percentage (%)
4: Rarely used DevOps	62	21
5: Sometimes used DevOps	97	32
6: Often used DevOps	82	28
7: Always used DevOps	55	19
Total	296	100

calculated with the help of the degree to which the goals were addressed) shown in Table 4. The first 20 parameters have been used as the set of inputs for the machine learning algorithms and the parameter effectiveness scores have been treated as output. The 296 participants' responses about each software were recorded in the survey. The general statistics such as mean and standard deviation against each software practice and goal addressed are mentioned in Table 5a and b respectively. Despite using limited sample size, our study that used ML to predict DevOps performance has revealed improved classification accuracy (Vabalas et al. 2019). We looked at the possibility that this bias may be brought on using validation techniques that do not adequately control overfitting. Our simulations demonstrate that K-fold Cross-Validation (CV) yields significantly skewed performance estimates for small sample sizes (Bar-Hillel 1979; Martínez-Mesa et al. 2014; Beck 2013).

A distribution graph is also plotted (Fig. 3) in order to ascertain the linearity of the software practices. It can be observed from Fig. 3 that for most of the practices; their values are more than 5 i.e., current dataset contains observations where the mentioned practices are heavily used. A correlation matrix is presented as a Heatmap in Fig. 4a to determine the degree of association among practices. Greener shades represent a higher correlation while brown/yellow colors show lesser association among practices. For a robust predictive model, dropping one feature from the pair of highly correlated features is necessary. Given that there is less correlation between the practices, the efficacy of DevOps deployment was estimated considering all of the practices. We have added the heat map for the identified 18 goals i.e., GA01 to GA18 as shown in Fig. 4b. The correlation between GA02 and GA03 is found to be 0.59, indicating that GA02 and GA03 are positively related to each other i.e., if the values of GA2 increase, the values of GA03 also increase with a correlation coefficient 0.59. Similarly, the GA02 is also found to be positively correlated with GA06, GA08, GA11, GA12, GA14, GA15, GA16, GA17 and GA18. This result shows that if GA02 is managed and controlled, the other goals could be achieved (i.e., GA3, GA6, GA8, GA11, GA12, GA14, GA15, GA16, GA17 and GA18) for effectively implementing DevOps practices in the software organizations. Similarly, the correlation between other goals is shown in Fig. 4b.

Table 4 Final version of dataset determined for DevOps to use in ML prediction algorithms

Input parameters																			Output parameters
20 Parameters (Use of DevOps + 19 Software practices)																			1 Parameter (Effectiveness score)
Use of DevOps	SP1	SP2	SP3	SP4	SP5	SP6	SP7	SP8	SP9	SP10	SP11	SP12	SP13	SP14	SP15	SP16	SP17	SP18	SP19	Effectiveness score
296 rows representing collected responses																				

Table 5 Statistics of values provided by participants

Software practices					
S. No	Software practices	Min value	Max value	Mean	Std. Deviation
SP1	Architecture specifications	1	7	5.061	1.587
SP2	Automated unit testing	1	7	5.632	1.316
SP3	Backlog management	1	7	5.794	1.403
SP4	Burndown charts	1	7	4.98	1.77
SP5	Code review	2	7	5.902	1.126
SP6	Coding standards	2	7	5.973	1.041
SP7	Collective code ownership	1	7	4.841	2.063
SP8	Continuous deployment	1	7	5.189	1.50
SP9	Continuous integration	1	7	5.743	1.328
SP10	Daily ctandup	1	7	5.732	1.584
SP11	Definition of done/ready	1	7	5.199	1.84
SP12	Expert/Team-based estimation	1	7	5.281	1.659
SP13	Formal estimation	1	7	4.939	1.716
SP14	Iteration/Sprint reviews	1	7	5.557	1.537
SP15	Limit work in progress	1	7	5.767	1.331
SP16	Refactoring	1	7	5.426	1.127
SP17	Release planning	2	7	5.753	1.225
SP18	Retrospectives	1	7	5.544	1.559
SP19	User stories	1	7	5.497	1.442

Table 5 (continued)

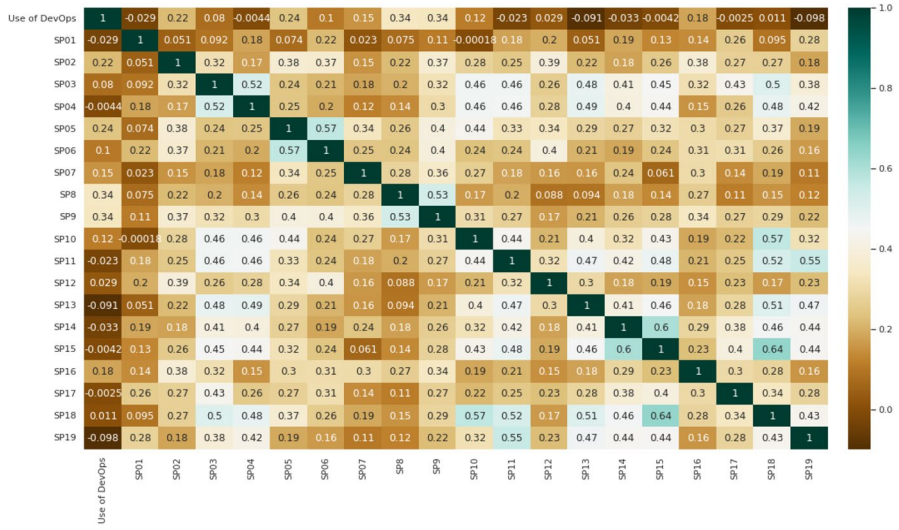
Goal addressed					
S. No	Goal addressed	Min value	Max value	Mean	Std. Deviation
GA01	Improved client involvement	2	10	7.035294	1.957822
GA02	Improved planning and estimation	1	10	6.426316	2.170593
GA03	Improved project monitoring and controlling	1	10	6.628049	2.165416
GA04	Improved reuse for project artifacts	1	10	6.308411	1.983
GA05	Improved tool support	1	10	6.090909	2.62168
GA06	Improved frequency of delivery to customers	1	10	7.42268	2.234144
GA07	Improved external product quality	1	10	6.924242	1.953604
GA08	Improved internal artifact quality	1	10	6.993865	2.112564
GA09	Improved knowledge transfer and learning	1	10	6.839416	2.139384
GA10	Improved employee satisfaction	1	10	6.907285	2.373046
GA11	Improved adaptability and flexibility of the process to react to change	1	10	7.404624	2.070946
GA12	Improved productivity	1	10	6.487923	2.304196
GA13	Improved staff education and development	1	10	7.039604	2.068433
GA14	Improved ability of the company to develop critical systems	1	10	6.493333	2.390145
GA15	Improved maturity of the company	1	10	6.515789	2.453236
GA16	Improved risk management	1	10	6.217742	2.074077
GA17	Improved return-on-investment cycles	1	10	6.357798	2.246491
GA18	Improved time-to-market	1	10	6.741176	2.389092

Fig. 3 Distribution of individual software practices

3.4 Machine learning based prediction

3.4.1 Prediction algorithms

Prediction algorithms are used to predict possible future outcomes (Filippetto et al. 2021). Predictions about the outcome of an impending event are made based on patterns of evidence. These algorithms analyze current and historical data in the generated model to predict possible outcomes. A predictive analysis model is revised at regular intervals to accommodate changes in the underlying data. Most prediction algorithms are fast enough to compute in real-time (Mair et al. 2000). The machine learning prediction algorithms require a set of inputs known as features along with corresponding outputs. There are a variety of prediction algorithms available to work with. Some of them are discussed below, which have also been used in the present study. In this study, we used three classification algorithms SVM, RF and ANN because they are among the well-established and most used algorithms. Also, many studies have established the superiority of these algorithms over others (Fernández-Delgado et al. 2014). Moreover, we considered specific properties of classifiers, such as SVM is less prone to overfitting and tends to perform better when the dataset size is small (Nadeem et al. 2016) and neural nets are considered universal function approximators (Nadeem et al. 2017).



(a) between practices

(b) between Goals

Fig. 4 Heatmap representing correlation

(a) Support vector machine: Support Vector Machine (SVM) is a supervised machine learning-based algorithm with the capability of classification, regression, and even outlier detection (Sheykhmousa et al. 2020). It is implemented with the aim of generating hyperplanes in n-dimensional space that classifies data points explicitly. SVM algorithm can be employed to solve both linear and

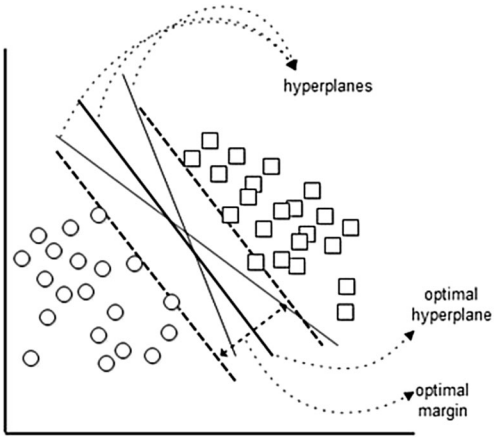


Fig. 5 Working principle of SVM

non-linear problems and also solves many practical problems.This gives effective results for cases where the number of dimensions exceeds the number of samples. SVM is considered less prone to overfitting and performs better even when the size of the available dataset is small. SVM tries to find a hyperplane that yields the largest margin between various classes as demonstrated in Fig. 5. Hence, it is also referred to as a large-margin classifier (Chauhan et al. 2019). SVM uses kernel functions that help modify existing data using complex transformation techniques and generate a hyperplane (decision boundary) to divide the samples into different classes.

(b) Artificial neural network: An artificial neural network (ANN) is a computational model that receives inputs into multiple processing elements and gives outputs with the help of predefined activation functions (Gupta and Raza 2019). It is named an artificial neural network because it mimics the way nerve cells work in the human brain. Artificial neural networks solve complex classification and approximation (prediction) problems through the process of learning from experience. Neural networks identify patterns in data to collect knowledge by identifying the dependency of outputs on corresponding input data. An ANN structure is a combination of simpler processing entities, arranged in layers, called neurons which are connected to share information (Abiodun et al. 2019). An ANN consists of one output layer, one input layer, and one or more hidden layers as depicted in Fig. 6.

(c) Random forest: The Random Forest algorithm is mainly used to solve classification and regression problems. It is named Random Forest because it is built on the foundation of a group of decision trees (Speiser et al. 2019). A random forest inserts multiple classifier decision trees on individual small sub-samples of the dataset.It uses the concept of averaging to control over-fitting and improve prediction accuracy. This algorithm includes a mechanism that maintains accuracy when a large chunk of data is missing. This can be seen as an aggregate/combination method which is more beneficial than a single decision tree as it minimizes over-fitting by equalizing the result, as shown in Fig. 7.

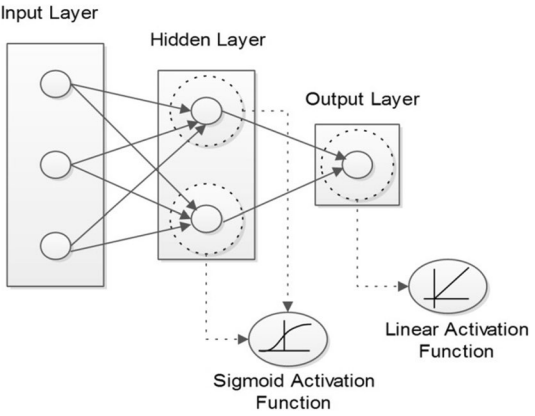


Fig. 6 ANN with three neurons in the input layer, two neurons in the hidden layer with sigmoid activation function, and one neuron in the output layer with linear activation function

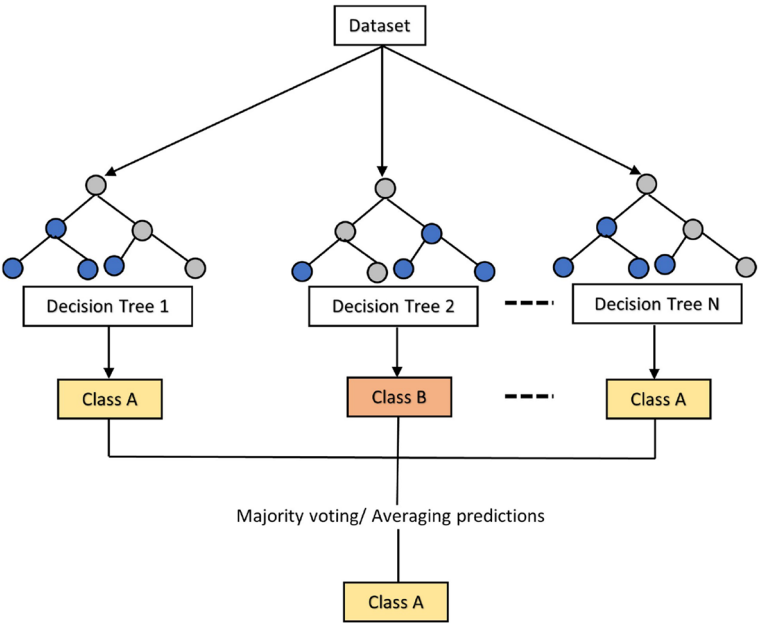


Fig. 7 A typical architecture of a Random Forest

3.4.2 Parameter tuning

Parameter tuning is an exercise of choosing the optimal set of parameters for learning algorithms (Mantovani et al. 2016). Machine learning models cannot learn from data alone but also need to provide parameters before getting into the training phase. Parameter tuning is a defined set of decisions for a machine learning model that has a direct impact on the training process and prediction results. In the present study, various models of the three algorithms were tested by changing the hyperparameter

values. The parameter tuning of SVM, ANN, and RF for training- testing and cross-validation are detailed in the following section.

(a) Parameter tuning in SVM: The effectiveness of Support Vector Machine significantly depends on the kernel function and parameters of the kernel function (Imbault and Lebart 2004). It maps the observations to some higher-dimensional feature space. There are several standard kernels for transforming observations into features, e.g., Polynomial Kernel, Linear Kernel, and Radial Kernel. The SVM was used along with all three kernels.
(b) Parameter tuning in ANN: In designing and training an ANN, parameters such as learning rate, the number of hidden layers, number of neurons in each layer, and epochs must be defined (Andreassen and Nachman 2020). The ANN was applied by making several combinations of training time, learning rate, and the number of hidden layers in the present study. We have applied multilayer perceptron (MLP) with (N = 5000,L = 0.3; H = 2), (N = 500; L = 0.1; H = 1), (N = 250; L = 0.1; H = 1), (N = 250; L = 0.3; H = 1), (N = 250; L = 0.1; H = 2), (N = 250; L = 0.3; H = 2), etc. where N = Number of epochs, L = Learning rate and H = Number of hidden layers.Out of these combinations, we have kept the results of two models.Thetwo accepted versions of the MLP were,one with a training time of 250, a learning rate of 0.1, and 1 hidden layer, and the other version with a training time of 250, a learning rate of 0.1, and 2hidden layers.
(c) Parameter tuning in RF: The parameters of the Random Forest Classifier that can be varied are the maximum depth of a tree, the number of trees in the forest, and the bootstrap (Probst et al. 2019). The algorithm was tried along with using different values for the maximum depth of the tree as 0, 1, 2, 3, 6, etc. The default case of Random Forest with a maximum depth = 0 and bootstrap = true was adopted.The default version is considered for use in the prediction of training- testing, and cross-validation results.

3.4.3 Training and testing

The prediction algorithms are trained and tested by the use of data related to DevOps (Putra et al. 2020). A software named WEKA was used to perform the evaluation and analysis.After removing outliers, missing values, and other redundancies where goals were not addressed, 296 instances were determined that were completely related to DevOps.The dataset for testing was selected randomly from all 296 cases of the dataset. The 60 responses were selected for testing and 236 remaining responses were consumed for training the results. These 236 were supplied to all the prediction models.

3.4.4 Cross-validation

Cross-validation is a method of data resampling used in tuning model parameters and estimating the prediction error of the model (Ramezan et al. Jan. 2019). It is useful in tackling overfitting when the dataset is small. We implemented cross-validation approach on all the models mentioned in the previous sectionby changing the

tuning parameter values and adapting the parameter to form the final model that has the minimum cross-validation error. We employed k-fold cross-validation approach where the entire learning set is divided into k disjoint subsets of equal size (Xiong et al. 2020). The whole set of 296 responses was used for cross-validation and it was divided into 10-folds.

3.4.5 Sensitivity analysis

Sensitivity analysis involves estimating the individual effects of each parameter on the output by varying the inputs to the system.If employed properly, it can reveal additional insights that would otherwise have been missed. To perform the sensitivity analysis, the input parameters were removed one by one from the dataset and the errors were recorded. If the error increases after removing the parameter, it suggests that the parameter has important in predicting the outcome accurately (Razavi et al. 2021). But if the increase in error was observed to be negligible, then the parameter is considered to be less significant. In our study, the available data were supplied to SVM algorithm with one missing parameter to perform sensitivity analysis.

3.4.6 Error rate

Error rates are used to evaluate the predictive power of prediction algorithms. Error rates such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE) are employed as performance indicators for the accuracy of predictions. The MAE shows the difference in absolute terms between their actual and expected values. The root mean square error (RMSE) is the average of all square errors for all possible pairings of actual and forecasted values.

4 Results

In this section, the results obtained through the data pre-processing, training–testing approach, cross-validation, and sensitivity analysis are presented and discussed.

4.1 Effectiveness predicted through prediction models.

As mentioned in earlier sections, we have used three different algorithms i.e., Support Vector Machine, Artificial Neural Network, and Random Forest for predicting the effectiveness of DevOps. The machine learning models were trained on the training datasetwith 236 records. Moreover, the accuracy of the models was tested using the testing data set of size 60. Moreover, the errors are estimated by comparing the actual effectiveness score and the predicted effectiveness score. Mean absolute error and root mean squared error were determined for training–testing, and cross-validation approaches.

Table 6 Evaluated values of errors predicted through training and testing

S. No	Prediction algorithms		Mean absolute error	Root mean squared error
1	SVM Poly kernel	Training	0.1165	0.1583
		Testing	0.1204	0.1488
2	SVM RBF kernel	Training	0.12	0.1625
		Testing	0.113	0.1388
3	ANN Learning rate = 0.1; Training time = 250; Hidden layer = 2	Training	0.1148	0.1425
		Testing	0.1177	0.1445
4	ANN Learning rate = 0.1; Training time = 250; Hidden layer = 1	Training	0.0751	0.0989
		Testing	0.1644	0.2137
5	Random Forest	Training	0.0481	0.0615
		Testing	0.1227	0.1478

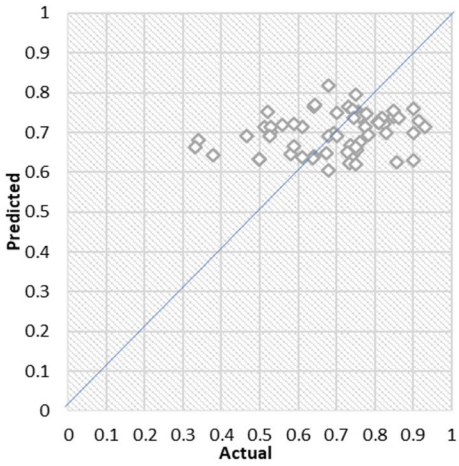


Fig. 8 Comparison of actual and predicted values of effectiveness score through SVM + RBF kernel

4.1.1 SVM model

SVM models, combined with three kernels, were evaluated for both training–testing, and cross-validation approaches. The SVM with polynomial kernel and RBF kernel produced results better than the SVM with linear kernel. Therefore, we provided SVM model results with these two kernels in Table 6. The SVM model with RBF Kernel provided the best accuracy (minimum error). For SVM + RBF kernel model, the individual values of the actual and predicted effectiveness scores related to each record are plotted in Fig. 8. Most of the points in the graph are close to the ideal line, which indicates that prediction is accurate enough. We have taken the mean of all the predicted effectiveness scores generated for each record and it is found to be

0.697. The mean of actual effectiveness scores calculated with the help of degrees of addressing the goals was 0.689. The difference between mean values actual (68.9%) and predicted effectiveness (69.7%) is very small, which indicates that the DevOps effectiveness evaluated by SVM + RBF kernel model is very close to its real value.

4.1.2 Artificial neural network model

Like SVM, MLP was also used to predict effectiveness scores for each record in the test dataset after training. MLP model having (N = 250; L = 0.1; H = 2) performed best among all MLP models. The predicted values, along with actual values for the same were used to plot the graph as shown in Fig. 9. The difference between the predicted effectiveness and actual effectiveness was observed to be greater than the SVM model, as the points representing the effectiveness are comparatively scattered. The average of all predicted effectiveness scores is determined as 0.656. The predicted effectiveness score generated from ANN with 2 hidden layers was far more than the actual effectiveness score of 0.689. This reflects a significant difference between the estimated effectiveness (65.6%) and the actual effectiveness (68.9%).

4.1.3 Random forest model

Similarly, Random Forest has been applied to estimate the effectiveness of DevOps. The accuracy of the predictions made by Random Forest with respect to effectiveness scores was examined by quantifying the difference between the predicted effectiveness and their actual effectiveness as determined through the model. Figure 10 represents the actual effectiveness and their respective predicted effectiveness. The points plotted in the graph appear to be more scattered than both SVM and MLP models. The mean score of all individual effectiveness scores estimated through the model was 0.641 (64.1%) in contrast to the average of the actual effectiveness scores i.e., 0.689 (68.9%).

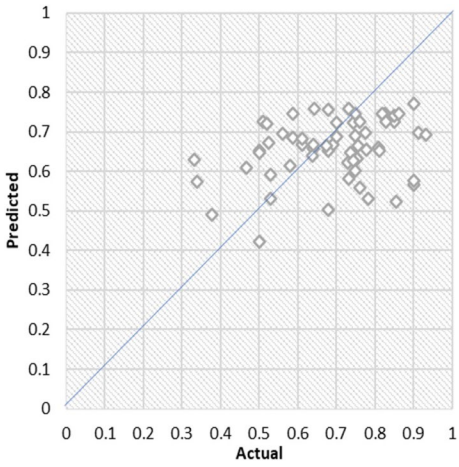


Fig. 9 Comparison of actual and predicted values of effectiveness score through MLP

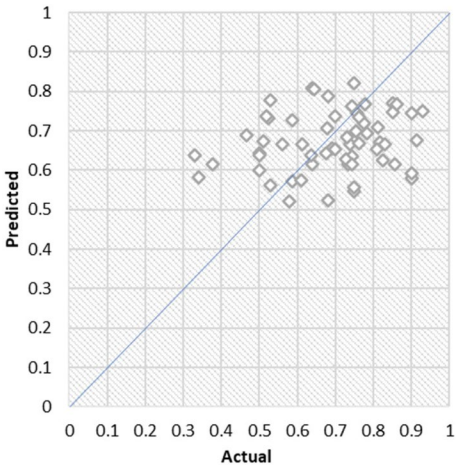


Fig. 10 Comparison of actual and predicted values of effectiveness score through RF

4.2 Performance evaluation of models

The predictive capabilities of all considered algorithms (i.e., SVM, ANN, and RF) were assessed by two statistical performance measures (Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE)). MAE indicates the absolute value of the difference between their actual and its predicted value.RMSE represents the mean value of all square errors for different pairs of actual and predicted values.

4.2.1 Evaluation under training and testing

The comparative values of both performance measures obtained during training and testing process are mentioned in Table 6. Both errors are placed in the Table 6 for all final five models of the three algorithms. The best variants of all the three prediction algorithms were selected to analyze their prediction capabilities. The top three types were SVM with RBF kernel, ANN with 2 hidden layers, and the default case of Random Forest. The Mean Absolute Error was used to compare their predictive powers in testing phase, which is depicted in Fig. 11. The error valuesof SVM with RBF kernel, ANN with 2 hidden layers, and Random Forest were 0.113, 0.1177, and 0.1227, respectively. A similar trend was observed with RMSE values as the respective errors were 0.1388**,** 0.1445 and 0. 1478.It can be also observed from Fig. 11 and Table 6 that SVM + RBF kernel performed better than the rest.

4.2.2 Evaluation under cross-validation

As discussed earlier, the tenfold cross-validation approach was also utilized due to limited-sized data. The same software WEKA was used to perform the evaluation and analysis. It was applied to the whole dataset of 296 instances. The configurations of algorithms were kept the same as mentioned in Table 6. The evaluation

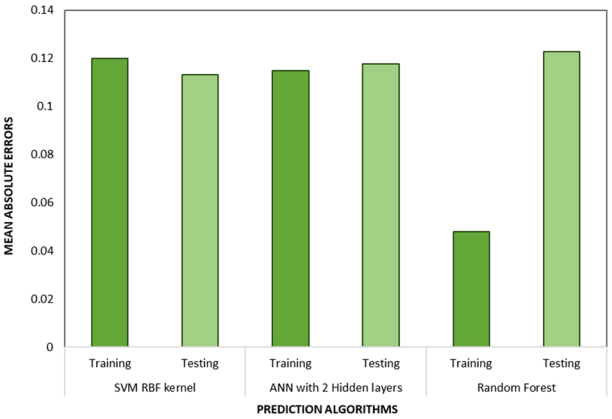


Fig. 11 Comparison of errors obtained by prediction algorithms in training and testing

Table 7 Estimated values of errors predicted through cross-validation

S. No	Prediction algorithms	Mean absolute error	Root mean squared error
1	SVM Polykernel	0.1305	0.1699
2	SVM RBFkernel	**0.1262**	**0.1605**
3	ANN Learning rate = 0.1; Training time = 250; Hidden layer = 2	0.1272	0.1676
4	ANN Learning rate = 0.1; Training time = 250; Hidden layer = 1	0.1738	0.2216
5	Random Forest	0.1264	0.165

Bold values represent the best error values among all the errors

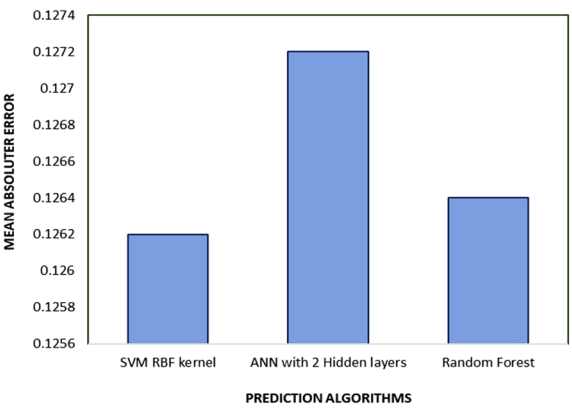


Fig. 12 Comparison of error for cross-validation through prediction algorithms

values of prediction in cross-validation are shown in Table 7 and Fig. 12. The results are similar to the training–testing approach i.e., SVM, combined with RBF Kernel, outperformed the models of other two algorithms. However, the random forest model performed better than MLP models, opposite to the results of training–testing approach.

4.3 Sensitivity analysis-based evaluation

Sensitivity analysis was also performed to determine the parameters having a higher contribution in the proposed model.To perform the sensitivity analysis, it was decided to drop software practice data values one by one from the dataset and use the remaining software practices were kept to predict the effectiveness of DevOps. The effectiveness score, determined after dropping the data for each software practice from the dataset, helps to derive the critical role each software practice plays in the effectiveness of DevOps.If dropped software practice leads to an increase in error between predicted effectiveness and actual effectiveness, the role of software practice in predicting effectiveness scores is considered as important. This process was repeated for all software practices for DevOps determined in our study. It was observed that when datasets were supplied in the absence of certain software practices, error has increased. An increase in errors indicates a decline in the efficiency of the prediction algorithm. The efficiency of prediction algorithms was reduced due to the removal of a necessary software practice. We considered SVM + RBF kernel for this analysis as it performed the best on both approaches. The errors determined after removal of software practices are shown in Table 8. The mean absolute error was determined as 0.126 when no software practice was removed. Figure 13 shows how errors varied when practices were removed one by one. Software practices whose deletion increased the prediction error were ten. The following ten software practices were found to be more important than the rest for implementing DevOps: Architecture Specifications,Code Review, Coding Standards, Continuous Deployment, Continuous Integration,Daily Standup, Expert/Team-Based Estimation, Formal Estimation, Iteration/Sprint Reviews and Limit work in Progress.

5 Discussion

This study was done to make software organizations aware of software practices that contribute to effective DevOps implementation and to propose a model that assesses the effectiveness of DevOps through the adoption of identified software practices.

5.1 DevOps practices and their effectiveness (RQ1)

The HELENA2 Dataset was utilized to extract the software practices that ease the process of DevOps implementation. A total of 19 software development practices as shown in Table 2 were identified. These practices were identified based on the frequency of use among DevOps users. The process of practice selection

Table 8 The MSE values to perform sensitivityanalysis

Parameter	Mean absolute error
All parameters	0.1260
(Missing) Use of DevOps	0.1271
(Missing) SP1-Architecture Specifications	0.1270
(Missing) SP2- Automated Unit Testing	0.1262
(Missing) SP3-Backlog Management	0.1261
(Missing)SP4-BurnDown Charts	0.1263
(Missing) SP5-Code review	0.1270
(Missing) SP6-Coding standards	0.1290
(Missing) SP7-Collective code ownership	0.1262
(Missing) SP8-Continuous deployment	0.1301
(Missing) SP9-Continuous integration	0.1282
(Missing) SP10-Daily Standup	0.1281
(Missing) SP11-Definition of done / ready	0.1263
(Missing) SP12-Expert/Team based estimation	0.1271
(Missing) SP13-Formal estimation	0.1270
(Missing) SP14-Iteration/Sprint Reviews	0.1270
(Missing) SP15-Limit Work InProgress	0.1273
(Missing) SP16-Refactoring	0.1261
(Missing) SP17-Release planning	0.1261
(Missing) SP18-Retrospectives	0.1262
(Missing) SP19-User Stories	0.1261

is explained in detail in Sect. 3.2.3. Software practices were selected that are often and always used by DevOps practitioners. The effectiveness of the identified practices in implementing DevOps is assessed based on effectiveness scores. Effectiveness scores associated with each record representing data from 19 software practices were calculated. The average effectiveness score of all identified DevOps related records was determined as 0. 689. When this dataset was performed on the prediction algorithm, the mean of all predicted effectiveness scores from SVM, ANN and RF were obtained to be 0.697, 0.656 and 0.671, respectively, which are close to the actual effectiveness scores. Actual and projected effectiveness results show that the identified practices are 65% to 70% effective in implementing DevOps. The contribution of each software in predicting the DevOps effectiveness was evaluated through the process of sensitivity analysis. The complete process of sensitivity analysis is detailed in Sect. 4.3. The role of 10 software practices such as Architecture Specifications, Code Review, Coding Standards, Continuous Deployment, Continuous Integration, Daily Standup, Expert/Team-Based Estimation, Formal Estimation, Iteration/Sprint Reviews, and Limit work in Progress have been determined as critical to DevOps implementations because their removal has resulted in increased mean absolute error.

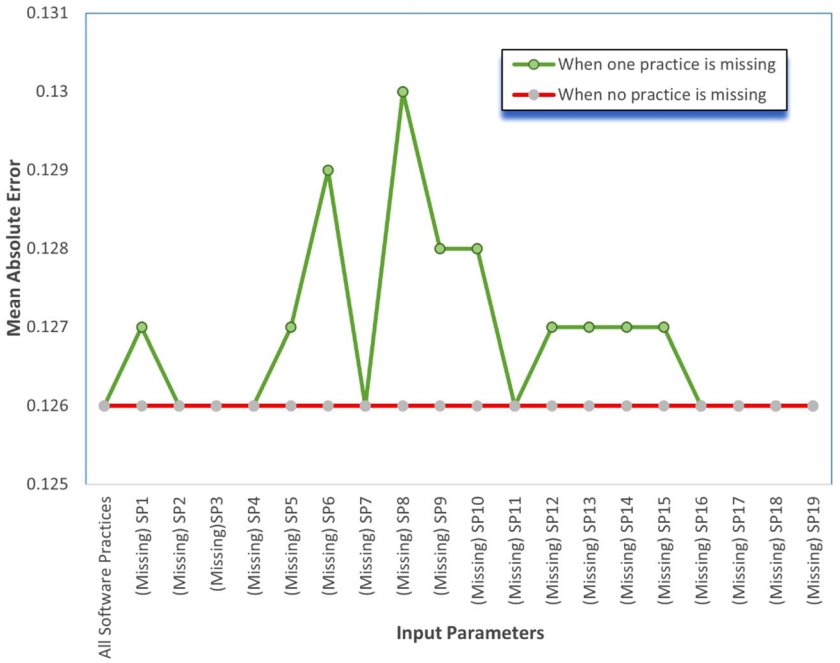


Fig. 13 Sensitivity analysis based on SVM RBF kernel method

5.2 Prediction model for DevOps effectiveness (RQ2 and RQ3)

The goal of this study was to propose a DevOps effectiveness model to predict the effectiveness of DevOps implementation which can ensure the software organizations for efficiently and effectively implement in the software organizations. For implementing DevOps culture in the software organizations, it is important to know which areas or practices are more significant to be focused on for successful implementation of DevOps. Moreover, this will ultimately help the organizations to reduce the risk of DevOps project failure. In order to achieve our objectives as listed in Sect. 3.1.2, we have developed a model to help find out the practices that are more significant in determining DevOps project effectiveness. The proposed models evaluated the effectiveness in terms of quantitative numbers that has the potential to address the level of DevOps within software organizations.

The model was developed using Support Vector Machine prediction algorithm with RBF kernel to evaluate the effectiveness. The models return an effectiveness score that is very close to the real-world effectiveness of the DevOps projects. The sensitivity analysis of the prediction model (Fig. 11) indicates that the SP6, SP8, SP9, SP10 are found to be the most significant DevOps practices that the organizations need to focus more on. This finding shows importance of SP8- (Continuous deployment) in Continuous Quality as the DevOps process ensures that there is a well-defined testing process is required at the Integration level and deployment level. This finding is consistent with the findings of the study conducted by Rathod

&Survey (Speiser et al. 2019). SP6: Coding standards was found to be the second most important practice in the prediction model that plays a significant role in evaluating the effectiveness. DevOps also promotes coding standardizations as it has a positive impact on product quality. Writing a standardized quality code requires a consistent effort of the software development teams to meet code quality goal (Mantovani et al. 2016). Hence, it is important for organizations to focus on code standardization for effectively implementing DevOps and delivering a quality product.

6 Limitations of the study

The present research is based on the results of HELENA2 survey conducted in 2017. This research does not cover other emerging practices, such as the use of microservices, as the study is based on the HELENA2 survey that was carried out much earlier.The study had over 200 data points that were used to collect feedback about 24 software methodologies and 36 software development practices.For our study, there are many data points left unusable in the dataset which have been ignored.The entire dataset collected 1467 responses which includes feedback about all 24 software development frameworks. The other frameworks included in the survey were Waterfall model, Crystal family, Iterative development, Kanban, Spiral model, etc. When responses were extracted from the entire dataset devoted solely to DevOps, the number of such responses was reduced to 467. Further out of 437 responses, only 296 responses have been used in the present study, and the remaining responses have been omitted in which participants did not provide feedback about the goals being addressed. There was a possibility that critical responses might be overlooked when restricting the dataset to DevOps-oriented data. The data of 296 responses related to DevOps was very less for the application of machine learning prediction algorithms.

7 Implications of the study

A machine learning-based model is proposed in this work with the goal to develop a model to evaluate the effectiveness of DevOps implementation in software organizations. The fact about machine learning algorithms is that the quality and quantity of data provided to machine learning prediction algorithms can affect the learning algorithm's predictive ability. The model proposed in this study can be considered reliable as it is based on the data collected by the HELENA2 dataset, which is a knowledge base for both DevOps researchers as well as industry practitioners. The proposed models have the potential to be used with confidence in software organizations to predict the effectiveness against the existing capabilities of the organizations. The model results provide a list of areas that need to be addressed by the organizations to improve their management and development strategies for delivering a quality product. The application of ML approaches in DevOps provides s knowledge base for DevOps researchers to implement ML techniques i.e., SVM, RF. Moreover, it could also provide an understanding of how DevOps effectiveness depends on the data of software practices. If the data used in prediction algorithms

increase in the future, this study could be improved to predict the effectiveness of implementing DevOps in software organizations.

8 Conclusion

The importance of DevOps in software development industry inspired us to develop a machine learning based model for predicting effectiveness of DevOps. Predicting the effectiveness of DevOps will help software development organizations to know their position in the DevOps journey.The effectiveness was estimated by machine learning algorithms through learning from previous data of most important software practices for implementing DevOps. The nineteen most important software practices were determined by the analysis of HELENA2 dataset. The adopted software practices were used widely for implementing DevOps as these practices were identified based on the frequency of use among DevOps users.The practices data was linked with the actual effectiveness score. Then, we have run the three different prediction algorithms such as Support Vector Machine, Artificial Neural Network, and Random Forest for predicting the effectiveness of DevOps inside software organization with the help of previous data. The data from these 19 software practices has resulted in improved effectiveness which is revealed from the results of various prediction algorithms.The predicted effectiveness determined by all the learning algorithms is very close to the actual effectiveness determined with the help of goals. But the analysis reveals that SVM with RBF kernel produced the best results as it has the least value of prediction errors. Moreover, we have also performed a sensitivity analysis using Support Vector Machines with RBF kernel. The results of sensitivity analysis explored that 10 software practices out of 19 DevOps-related software practices were determined as more significant for DevOps implementation.

Author contributions Conceptualization, A. B: done Data curation, Investigation, Methodology, Validating, Formal analysis.; B, C: Methodology, Reviewing & Editing, Validating; Reviewing, Editing, Formal Analysis.

Data availability The dataset analyzed during the current study is available using following link (https://www.researchgate.net/publication/329246439_HELENA_Stage_2_Results) and is referenced in the manuscript (Kuhrmann et al. 2018).

Declarations

Conflict of interest The authors have no competing interests to declare that are relevant to the content of this article.

References

Abiodun, O.I., et al.: Comprehensive review of artificial neural network applications to pattern recognition. IEEE Access **7**, 158820–158846 (2019). https://doi.org/10.1109/ACCESS.2019.2945545

Akbar, M.A., Mahmood, S., Shafiq, M., Alsanad, A., Alsanad, A.A.-A., Gumaei, A.: Identification and prioritization of DevOps success factors using fuzzy-AHP approach. Soft Comput. (2020). https://doi.org/10.1007/s00500-020-05150-w

Albuquerque, A.B., Cruz, V.L.: Implementing DevOps in Legacy Systems, pp 143–161. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-00184-1_14

Almeida, F., Simões, J., Lopes, S.: Exploring the benefits of combining DevOps and agile. Future Internet **14**(2), 63 (2022). https://doi.org/10.3390/fi14020063

Amaro, R.M.D., Pereira, R., da Silva, M.: Capabilities and practices in DevOps: a multivocal literature review. IEEE Trans. Softw. Eng. **49**, 883–901 (2022). https://doi.org/10.1109/TSE.2022.3166626

Amershi, S.: et al. Software engineering for machine learning: a case study. In: 2019 IEEE/ACM 41st International Conference on Software Engineering: Software Engineering in Practice (ICSE-SEIP), 2019, pp 291–300. doi: https://doi.org/10.1109/ICSE-SEIP.2019.00042.

Anandya, R., Raharjo, T., Suhanto, A.: Challenges of DevOps implementation: a case study from technology companies in Indonesia. In: 2021 International Conference on Informatics, Multimedia, Cyber and Information System (ICIMCIS, 2021, pp 108–113. doi: https://doi.org/10.1109/ICIMCIS53775.2021.9699240.

Andreassen, A., Nachman, B.: Neural networks for full phase-space reweighting and parameter tuning. Phys. Rev. D **101**(9), 091901 (2020). https://doi.org/10.1103/PhysRevD.101.091901

Angara, J., Gutta, S., Prasad, S.: DevOps with continuous testing architecture and its metrics model. In: Recent Findings in Intelligent Computing Techniques, pp 271–281. Springer, Singapore (2018)

Azad, N.: Understanding DevOps critical success factors and organizational practices. IEEE/ACM Int. Workshop Softw. -Intens. Bus. (IWSiB) **2022**, 83–90 (2022). https://doi.org/10.1145/3524614.3528627

Badshah, S., Khan, A.A., Khan, B.: Towards Process Improvement in DevOps: A Systematic Literature Review. In: Proceedings of the Evaluation and Assessment in Software Engineering, 2020, pp 427–433. doi: https://doi.org/10.1145/3383219.3383280

Bar-Hillel, M.: The role of sample size in sample evaluation. Organ. Behav. Hum. Perform. **24**(2), 245–257 (1979). https://doi.org/10.1016/0030-5073(79)90028-X

Beck, T.W.: The importance of A Priori sample size estimation in strength and conditioning research. J. Strength Cond. Res. **27**(8), 2323–2337 (2013). https://doi.org/10.1519/JSC.0b013e318278eea0

Benni, B., Blay-Fornarino, M., Mosser, S., Precioso, F., Jungbluth, G.: When DevOps meets meta-learning: a portfolio to rule them all. In 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C), 2019, pp 605–612. doi: https://doi.org/10.1109/MODELS-C.2019.00092

Bijwe, A., Shankar, P.: Analysis of factors that improve reliability and effectiveness of DevOps culture in developing connected devices. SSRN Electron. J. (2022). https://doi.org/10.2139/ssrn.4091401

Brand, M., Tiberius, V., Bican, P.M., Brem, A.: Agility as an innovation driver: towards an agile front end of innovation framework. RMS **15**(1), 157–187 (2021). https://doi.org/10.1007/s11846-019-00373-0

Bruneliere, H., et al.: AIDOaRt: AI-augmented automation for DevOps, a model-based framework for continuous development in cyber-physical systems. Microprocess. Microsyst. **94**, 104672 (2022). https://doi.org/10.1016/j.micpro.2022.104672

Çalikli, G., Staron, M, Meding, M.: Measure early and decide fast: transforming quality management and measurement to continuous deployment. In Proceedings of the 2018 International Conference on Software and System Process, 2018, pp 51–60. doi: https://doi.org/10.1145/3202710.3203156.

Callanan, M., Spillane, A.: DevOps: making it easy to do the right thing. IEEE Softw. **33**(3), 53–59 (2016). https://doi.org/10.1109/MS.2016.66

Castellanos, C., Varela, C.A., Correal, D.: ACCORDANT: a domain specific-model and DevOps approach for big data analytics architectures. J. Syst. Softw. **172**, 110869 (2021). https://doi.org/10.1016/j.jss.2020.110869

Chakraborty Bapi, S., Karthikeyan, A.: Continuous Monitoring and Changes. In: Understanding Azure Monitoring: Includes IaaS and PaaS Scenarios, Berkeley, CA: Apress, 2019, pp 205–216. doi: https://doi.org/10.1007/978-1-4842-5130-0_6.

Chauhan, V.K., Dahiya, K., Sharma, A.: Problem formulations and solvers in linear SVM: a review. Artif. Intell. Rev. **52**(2), 803–855 (2019). https://doi.org/10.1007/s10462-018-9614-6

Crowley, Catherine, Louise Veling, Linda Beckett, Graeme Clarke, Eamon Kelleher, John McHale, Laura McQuillan, and Shaun Percival. "A DevOps Capability-The IVI DevOps Effectiveness Assessment." (2018).

Dehgani, R., Jafari Navimipour, N.: The impact of information technology and communication systems on the agility of supply chain management systems. Kybernetes **48**(10), 2217–2236 (2019). https://doi.org/10.1108/K-10-2018-0532

Dörnenburg, E.: The path to DevOps. IEEE Softw **35**(5), 71–75 (2018). https://doi.org/10.1109/MS.2018.290110337

Erich, F.M.A., Amrit, C., Daneva, M.: A qualitative study of DevOps usage in practice. J. Software: Evol. Process (2017). https://doi.org/10.1002/smr.1885

Erich, F.M.A., Amrit, C., Daneva, M.: A qualitative study of DevOps usage in practice. J. Software: Evol. Process (2017b). https://doi.org/10.1002/smr.1885

Faustino, J., Adriano, D., Amaro, R., Pereira, R., da Silva, M.M.: <scp>DevOps</scp> benefits: a systematic literature review. Softw Pract Exp **52**(9), 1905–1926 (2022). https://doi.org/10.1002/spe.3096

Fernández-Delgado, M., Cernadas, E., Barro, S., Amorim, D.: Do we need hundreds of classifiers to solve real world classification problems? J. Mach. Learn. Res. **15**(1), 3133–3181 (2014)

Filippetto, A.S., Lima, R., Barbosa, J.L.V.: A risk prediction model for software project management based on similarity analysis of context histories. Inf. Softw. Technol. **131**, 106497 (2021). https://doi.org/10.1016/j.infsof.2020.106497

Fitzgerald, B., Stol, K.-J.: Continuous software engineering: a roadmap and agenda. J. Syst. Softw. **123**, 176–189 (2017). https://doi.org/10.1016/j.jss.2015.06.063

Forsgren, N., Kersten, M.: DevOps metrics. Commun. ACM **61**(4), 44–48 (2018). https://doi.org/10.1145/3159169

Gall, M., Pigni, F.: Taking DevOps mainstream: a critical review and conceptual framework. Eur. J. Inf. Syst. **31**(5), 548–567 (2022). https://doi.org/10.1080/0960085X.2021.1997100

Gheorghe-Pop, I.-D., Tcholtchev, N., Ritter, T., Hauswirth, M.: Quantum DevOps: towards reliable and applicable NISQ Quantum Computing. In 2020 IEEE Globecom Workshops (GC Wkshps, 2020, pp 1–6. doi: https://doi.org/10.1109/GCWkshps50303.2020.9367411

Gupta, T.K., Raza, K.: Optimization of ANN architecture: a review on nature-inspired techniques. In: Machine Learning in Bio-Signal Analysis and Diagnostic Imaging, pp 159–182. Elsevier, New York (2019). https://doi.org/10.1016/B978-0-12-816086-2.00007-2

Gupta, V., Kapur, P.K., Kumar, D.: Modeling and measuring attributes influencing DevOps implementation in an enterprise using structural equation modeling. Inf. Softw. Technol. **92**, 75–91 (2017). https://doi.org/10.1016/j.infsof.2017.07.010

Heine K.M.: *Predicting DevOps Effectiveness in Information Technology (IT) Projects* (Doctoral dissertation, The George Washington University).

Hemon, A., Lyonnet, B., Rowe, F., Fitzgerald, B.: From agile to DevOps: smart skills and collaborations. Inf. Syst. Front. **22**(4), 927–945 (2020). https://doi.org/10.1007/s10796-019-09905-1

Hemon, A., Fitzgerald, B., Lyonnet, B., Rowe, F.: Innovative Practices for knowledge sharing in large-scale DevOps. IEEE Softw. **37**(3), 30–37 (2020). https://doi.org/10.1109/MS.2019.2958900

Imbault, F., Lebart, K.: A stochastic optimization approach for parameter tuning of support vector machines. In: Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004., 2004, vol. 4, pp 597–600. doi: https://doi.org/10.1109/ICPR.2004.1333843.

Ivanova, A., Ivanova, P.: Data analytics for devops effectiveness (2018)

Karamitsos, I., Albarhami, S., Apostolopoulos, C.: Applying DevOps practices of continuous automation for machine learning. Information **11**(7), 363 (2020). https://doi.org/10.3390/info11070363

Khan, A.A., Shameem, M.: Multicriteria decision-making taxonomy for DevOps challenging factors using analytical hierarchy process. J. Softw.: Evol. Process (2020). https://doi.org/10.1002/smr.2263

Khan, A.A., Shameem, M., Nadeem, M., Akbar, M.A.: Agile trends in Chinese global software development industry: Fuzzy AHP based conceptual mapping. Appl. Soft Comput. **102**, 107090 (2021). https://doi.org/10.1016/j.asoc.2021.107090

Khan, M.S., Khan, A.W., Khan, F., Khan, M.A., Whangbo, T.K.: Critical challenges to adopt DevOps culture in software organizations: a systematic review. IEEE Access **10**, 14339–14349 (2022). https://doi.org/10.1109/ACCESS.2022.3145970

Kirk, D.: Exploring task equivalence for software engineering practice adaptation and replacement. In: Proceedings of the 2022 ACM SIGPLAN International Symposium on New Ideas, New Paradigms, and Reflections on Programming and Software, 2022, pp 33–47. doi: https://doi.org/10.1145/3563835.3567656.

Klünder, J. et al.: Determining context factors for hybrid development methods with trained models. In: Proceedings of the International Conference on Software and System Processes, 2020, pp 61–70. doi: https://doi.org/10.1145/3379177.3388898.

Kuhrmann, M., Tell, P., Klünder, J., Hebig, R., Licorish, S., MacDonell, S.: Helena stage 2 results. ResearchGate, 2018

Lazuardi, M., Raharjo, T., Hardian, B., Simanungkalit, T.: Perceived benefits of DevOps implementation in organization: a systematic literature review. In 2021 10th International Conference on Software and Information Engineering (ICSIE), pp 10–16 (2021). doi: https://doi.org/10.1145/3512716.3512718.

Leite, L., Rocha, C., Kon, F., Milojicic, D., Meirelles, P.: A survey of DevOps concepts and challenges. ACM Comput. Surv. **52**(6), 1–35 (2019). https://doi.org/10.1145/3359981

Lin, B., Cassee, N., Serebrenik, A., Bavota, G., Novielli, N., Lanza, M.: Opinion mining for software development: a systematic literature review. ACM Trans. Softw. Eng. Methodol. **31**(3), 1–41 (2022). https://doi.org/10.1145/3490388

Liu, L., Xie, D., Cheng, Y., Li, G.: Architecture Scheme of DevOps for Cross Network and Multiple Environment Collaboration. In The 5th International Conference on Computer Science and Application Engineering, Oct. 2021, pp 1–5. doi: https://doi.org/10.1145/3487075.3487116

Luz, W.P., Pinto, G., Bonifácio, R.: Adopting DevOps in the real world: a theory, a model, and a case study. J. Syst. Software **157**, 110384 (2019). https://doi.org/10.1016/j.jss.2019.07.083

Lwakatare, L.E., et al.: DevOps in practice: a multiple case study of five companies. Inf. Softw. Technol. **114**, 217–230 (2019). https://doi.org/10.1016/j.infsof.2019.06.010

Lwakatare, L.E., et al.: DevOps in practice: a multiple case study of five companies. Inf Softw Technol **114**, 217–230 (2019). https://doi.org/10.1016/j.infsof.2019.06.010

Lwakatare, L.E., Crnkovic, I., Bosch, J.: DevOps for AI—challenges in development of AI-enabled applications. In 2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM), 2020, pp 1–6. doi: https://doi.org/10.23919/SoftCOM50211.2020.9238323

Macarthy, R.W., Bass, J.M.: An empirical taxonomy of DevOps in practice. In: 2020 46th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2020, pp 221–228. doi: https://doi.org/10.1109/SEAA51224.2020.00046.

Mair, C., et al.: An investigation of machine learning based prediction systems. J. Syst. Softw. **53**(1), 23–29 (2000). https://doi.org/10.1016/S0164-1212(00)00005-4

Mantovani, R.G., Horvath, T., Cerri, R., Vanschoren, J., de Carvalho, A.C.P.L.F.: Hyper-Parameter Tuning of a Decision Tree Induction Algorithm. In 2016 5th Brazilian Conference on Intelligent Systems (BRACIS), 2016, pp 37–42. doi: https://doi.org/10.1109/BRACIS.2016.018.

Marijan, D., Gotlieb, A., Liaaen, M.: A learning algorithm for optimizing continuous integration development and testing practice. Softw. Pract. Exp **49**(2), 192–213 (2019). https://doi.org/10.1002/spe.2661

Marijan, D., Liaaen, M., Sen, S.: DevOps improvements for reduced cycle times with integrated test optimizations for continuous integration. In: 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), 2018, pp 22–27. doi: https://doi.org/10.1109/COMPSAC.2018.00012

Marrero, L., Astudillo, H.: DevOps-RAF: an assessment framework to measure DevOps readiness in software organizations. In: 2021 40th International Conference of the Chilean Computer Science Society (SCCC), 2021, pp 1–8. doi: https://doi.org/10.1109/SCCC54552.2021.9650363.

Martínez-Mesa, J., González-Chica, D.A., Bastos, J.L., Bonamigo, R.R., Duquia, R.P.: Sample size: How many participants do I need in my research? An. Bras. Dermatol. **89**(4), 609–615 (2014). https://doi.org/10.1590/abd1806-4841.20143705

Mirina, M., Mario, J., Negrete, J.: Proposal to Avoid Issues in the DevOps Implementation: A Systematic Literature Review. In: New Knowledge in Information Systems and Technologies, 2019, pp 666–677.

Morales, J.A., Yasar, H., Volkman, A.: Implementing DevOps practices in highly regulated environments. In: Proceedings of the 19th International Conference on Agile Software Development: Companion, 2018. doi: https://doi.org/10.1145/3234152.3234188.

Mumbarkar, P., Prasad, S.: Adopting DevOps: capabilities, practices, and challenges faced by organizations. PAIP Conf. Proc. **2022**, 030029 (2022). https://doi.org/10.1063/5.0110594

Nadeem, M., Banka, H., Venugopal, R.: SVM-based predictive modelling of wet pelletization using experimental and GA-based synthetic data. Arab. J. Sci. Eng. **41**(3), 1053–1065 (2016). https://doi.org/10.1007/s13369-015-1979-0

Nadeem, M., Banka, H., Venugopal, R.: Estimation of pellet size and strength of limestone and manganese concentrate using soft computing techniques. Appl. Soft Comput. **59**, 500–511 (2017). https://doi.org/10.1016/j.asoc.2017.06.005

Narang, P., Mittal, P.: Performance assessment of traditional software development methodologies and DevOps automation culture. Eng. Technol. Appl. Sci. Res. **12**(6), 9726–9731 (2022). https://doi.org/10.48084/etasr.5315

Nogueira, A.F., Ribeiro, J.C.B., Zenha-Rela, M.A., Craske, A.: Improving La redoute's CI/CD Pipeline and DevOps processes by applying machine learning techniques. In 2018 11th International Conference on the Quality of Information and Communications Technology (QUATIC), 2018, pp 282–286. doi: https://doi.org/10.1109/QUATIC.2018.00050.

Pianini, D., Neri, A.: Breaking down monoliths with microservices and DevOps: an industrial experience report. In: 2021 IEEE International Conference on Software Maintenance and Evolution (ICSME), Sep. 2021, pp 505–514. doi: https://doi.org/10.1109/ICSME52107.2021.00051.

Prado Lima, J.A., Vergilio, S.R.: Test case prioritization in continuous integration environments: a systematic mapping study. Inf. Softw. Technol. **121**, 106268 (2020). https://doi.org/10.1016/j.infsof.2020.106268

Probst, P., Wright, M.N., Boulesteix, A.: Hyperparameters and tuning strategies for random forest. WIREs Data Mining Knowl. Discov. (2019). https://doi.org/10.1002/widm.1301

Putra, T.A., Rufaida, S.I., Leu, J.-S.: Enhanced skin condition prediction through machine learning using dynamic training and testing augmentation. IEEE Access **8**, 40536–40546 (2020). https://doi.org/10.1109/ACCESS.2020.2976045

Rafi, S., Akbar, M.A., Yu, W., Alsanad, A., Gumaei, A., Sarwar, M.U.: Exploration of DevOps testing process capabilities: an ISM and fuzzy TOPSIS analysis. Appl. Soft Comput. **116**, 108377 (2022). https://doi.org/10.1016/j.asoc.2021.108377

Rahman, A., Mahdavi-Hezaveh, R., Williams, L.: A systematic mapping study of infrastructure as code research. Inf. Softw. Technol. **108**, 65–77 (2019). https://doi.org/10.1016/j.infsof.2018.12.004

Ramezan, C.A., Warner, T.A., Maxwell, A.E.: Evaluation of sampling and cross-validation tuning strategies for regional-scale machine learning classification. Remote Sens. **11**(2), 185 (2019). https://doi.org/10.3390/rs11020185

Razavi, S., et al.: The future of sensitivity analysis: an essential discipline for systems modeling and policy support. Environ. Model. Software **137**, 104954 (2021). https://doi.org/10.1016/j.envsoft.2020.104954

Rodríguez, P., Mäntylä, M., Oivo, M., Lwakatare, L.E., Seppänen, P., Kuvaja, P.: Advances in computersusing agile and lean processes for software development. In: Advances in Computers, pp 135–224. Elsevier, New York (2019)

Sabharwal, N., Bhardwaj, G.: Hands-on AIOps. Apress, Berkeley (2022). https://doi.org/10.1007/978-1-4842-8267-0

Saidani, I., Ouni, A., Mkaouer, M.W., Palomba, F.: On the impact of continuous integration on refactoring practice: an exploratory study on TravisTorrent. Inf Softw Technol **138**, 106618 (2021). https://doi.org/10.1016/j.infsof.2021.106618

Samarawickrama, S.S., Perera, I.: Continuous scrum: a framework to enhance scrum with DevOps. Seventeenth Int. Conf. Adv. ICT Emerg. Reg. (ICTer) **2017**, 1–7 (2017). https://doi.org/10.1109/ICTER.2017.8257808

Senapathi, M., Buchan, J., Osman, H.: DevOps capabilities, practices, and challenges: insights from a case study. In Proceedings of the 22nd International Conference on Evaluation and Assessment in Software Engineering 2018, 2018, pp 57–67. doi: https://doi.org/10.1145/3210459.3210465.

Shahin, M., Ali Babar, M., Zhu, L.: Continuous integration, delivery and deployment: a systematic review on approaches, tools, challenges and practices. IEEE Access **5**, 3909–3943 (2017). https://doi.org/10.1109/ACCESS.2017.2685629

Shameem, M.: A systematic literature review of challenges factors for implementing DevOps practices in software development organizations: a development and operation teams perspective. In: Evolving Software Processes, pp 187–199. Wiley, New York (2022). https://doi.org/10.1002/9781119821779.ch9

Shameem, M., Kumar, R.R., Nadeem, M., Khan, A.A.: Taxonomical classification of barriers for scaling agile methods in global software development environment using fuzzy analytic hierarchy process. Appl. Soft Comput. **90**, 106122 (2020). https://doi.org/10.1016/j.asoc.2020.106122

Shameem, M., Nadeem, M., Zamani, A.T.: Genetic algorithm based probabilistic model for agile project success in global software development. Appl. Soft Comput. **135**, 109998 (2023). https://doi.org/10.1016/j.asoc.2023.109998

Sheykhmousa, M., Mahdianpari, M., Ghanbari, H., Mohammadimanesh, F., Ghamisi, P., Homayouni, S.: Support vector machine versus random forest for remote sensing image classification: a meta-analysis and systematic review. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. **13**, 6308–6325 (2020). https://doi.org/10.1109/JSTARS.2020.3026724

Smeds, J., Nybom, K., Porres, I.: DevOps: A definition and perceived adoption impediments. 2015, pp 166–177. doi: https://doi.org/10.1007/978-3-319-18612-2_14.

Soni, M.: End to end automation on cloud with build pipeline: the case for DevOps in insurance industry, continuous integration, continuous testing, and continuous delivery. IEEE Int. Conf. Cloud Comput. Emerg. Markets (CCEM) **2015**, 85–89 (2015). https://doi.org/10.1109/CCEM.2015.29

Speiser, J.L., Miller, M.E., Tooze, J., Ip, E.: A comparison of random forest variable selection methods for classification prediction modeling. Expert Syst. Appl. **134**, 93–101 (2019). https://doi.org/10.1016/j.eswa.2019.05.028

Stahl, D., Martensson, T., Bosch, J.: Continuous practices and devops: beyond the buzz, what does it all mean?. In 2017 43rd Euromicro Conference on Software Engineering and Advanced Applications (SEAA), 2017, pp 440–448. doi: https://doi.org/10.1109/SEAA.2017.8114695.

Subramanya, R., Sierla, S., Vyatkin, V.: From DevOps to MLOps: overview and application to electricity market forecasting. Appl. Sci. **12**(19), 9851 (2022). https://doi.org/10.3390/app12199851

Vabalas, A., Gowen, E., Poliakoff, E., Casson, A.J.: Machine learning algorithm validation with a limited sample size. PLoS One **14**(11), e0224365 (2019). https://doi.org/10.1371/journal.pone.0224365

Wang, Z., Shi, M., Li, C.: An intelligent DevOps platform research and design based on machine learning. In 2020 Eighth International Conference on Advanced Cloud and Big Data (CBD), Dec. 2020, pp 42–47. doi: https://doi.org/10.1109/CBD51900.2020.00017

Xiong, Z., Cui, Y., Liu, Z., Zhao, Y., Hu, M., Hu, J.: Evaluating explorative prediction power of machine learning algorithms for materials discovery using k-fold forward cross-validation. Comput. Mater. Sci. **171**, 109203 (2020). https://doi.org/10.1016/j.commatsci.2019.109203