JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Improving La Redoute's CI/CD Pipeline and DevOps Processes by Applying Machine Learning Techniques

Dhaya Sindhu Battina
Sr. Data Engineer & Department of Information Technology
USA

Abstract— *This research paper explored how machine learning can be leveraged in improving CI/CD Pipeline and DevOps Processes. As a result of the intrinsic complexity of software creation and maintenance - not just in terms of technical complexity, but also from a human standpoint - some obstacles may be handled as learning problems. Software processes and products may benefit from machine learning approaches by gaining insight into tactics that can lead to better quality [1]. An ongoing study area predicts how likely something will be to fail due to a defect. Companies may increase their company value by operating in an agile mode, according to DevOps rules, which allows for more rapid communication, decision-making, and problem-solving [1]. It is described in this article how La Redoute's IT department is doing continuing research into the use of machine learning methods to increase the performance of tools and methodologies inside the DevOps pipeline.*

Keywords: *DevOps, CI, CD, machine learning, CI/CD Pipeline, software development*

I. INTRODUCTION

Machine learning (ML) is rapidly becoming a basic method in the field of data science, with applications ranging from addressing challenging real-world issues to revolutionizing enterprises and generating value in specific fields. So many data science teams are working in the scientific and machine learning communities to increase the entire company value by employing descriptive and predictive models, which is a common goal. As a result, teams of machine learning data scientists and machine learning operations engineers are researching ways to apply DevOps ideas to the machine learning systems they are researching. Developed in the context of software and systems engineering, DevOps [1] refers to a collection of processes and technologies. It is possible to describe software engineering as a field that is devoted to the development of tools and procedures that enable the building and usage of complex software systems. Instead of developing programs, data science focuses on the analysis of large amounts of data and concluding it. On the other hand, agile [2] is an iterative methodology that emphasizes teamwork, customer input, and short, frequent releases. A cross-functional group of developers and technicians, known as a DevOps or Agile team, may consistently improve operations by supporting the company's business plan using DevOps and Agile. The strategic goal of DevOps is to look for ways to improve service quality and features in a way that meets the demands of their clients [3]. Manual procedures for the delivery of the ML pipeline model have been established by the ML data scientists and the ML operations engineering teams. Owing to the reliance on the collection of data, preparations, and preprocessing, visualization tools, validating, and testing, this technique is likely to yield unexpected findings due to the reliance on these processes [3]. Furthermore, this strategy results in the conclusion that there is no obvious benefit to using our manual methodology for ML projects over other approaches. Aside from the significant operating expenses and delays associated with this ML manual pipeline approach, this method has a negative impact on the revenue or quality image of the business in terms of quality outcomes as well.

II. PROBLEM STATEMENT

The main problem that this paper will address is to explore the challenges and mitigation strategies when adopting DevOps during software development. Development and operations are sometimes at odds while providing meaningful products to consumers throughout the software development lifecycle. DevOps, an essential and rapidly evolving idea, is offered as a means of resolving the tension that exists between the development team and the operations team [3]. DevOps is becoming more popular among businesses and organizations. As a relatively new concept, DevOps necessitates an awareness of the potential problems as well as viable mitigation techniques.

LITERATURE REVIEW

A. CI/CD for Software Development Applications

When it comes to DevOps, continuous integration and continuous delivery (CI/CD) are practices that enable organizations to produce code and deploy it fast to production, whether it's for a client or an application. Thus, an automated pipeline is created, which allows you to develop code and have it ready for production in a short time. In software development, continuous integration and continuous delivery (CI/CD) involves many stages: First, one starts with a product request and a plan, then they code it, build it, and run it through a round of testing [4]. After the testing process is done, the system will switch from continuous integration) to continuous deployment. It starts with specifying the release, then deploys, operates, and monitors the application continuously.

Modeling is a continual process of updating and reassessing, identifying faults in the model, and going back to make changes to the model in response to new data [4]. This process is repeated over and over again. In addition, it automates the machine learning workflow (including model construction, testing, and deployment), which eliminates the need for software engineers to interfere in the process, rendering it more available to a larger audience and less prone to human error, since the feedback loop ensures that the models' precision and performance are always improving [5].

B. The components of a continuous integration/continuous delivery pipeline

A CI/CD pipeline may seem to be unnecessary overhead, but it is not. It is, in essence, a runnable specification of the processes that any developer must do to release a new version of a software product to the public [6]. Engineers would still have to conduct these stages manually, which would result in their being significantly less efficient in the devoid of an automated pipeline. The majority of software deployments go through some standard phases, shown in fig I below.

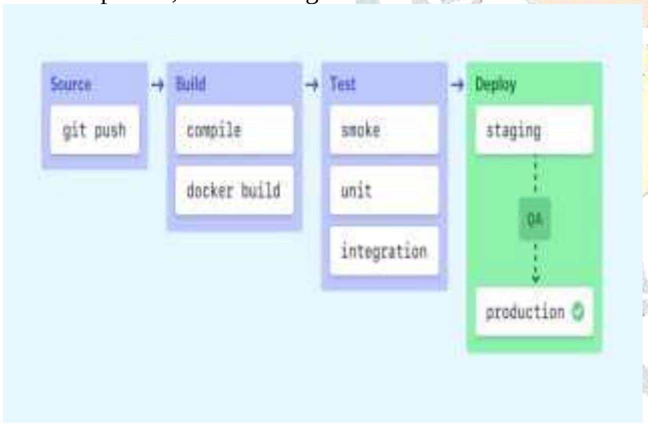

Fig i: Stages of a CI/CD pipeline

In most cases, failure at each level results in a message being sent to the concerned developers by email, Slack, or other means. If this is not the case, the whole team will be notified following each successful deployment to the production environment [6,7]. A pipeline might begin with a very basic design. A Go project pipeline that compiles the code, checks for code style and performs automated tests in two parallel processes is shown below.

C. CI/CD Best Practices For DevOps Teams

DevOps is the method through which teams collaborate with people, processes, and technology to provide value to their customers. Rather than focusing just on getting their products out the door, teams in a DevOps lifecycle must produce value in a timely, safe, and repeatable way to be successful. Continuous integration and continuous delivery (CI/CD) assist cross-functional teams in the performance of their tasks by automating, regulating, extending the life of, and securing the software delivery cycle[7]. This blog article will address continuous integration and continuous delivery (CI/CD) in the context of DevOps techniques, as well as the best CI/CD strategies to deploy for your DevOps teams.

D. How ML techniques improve CI/CD pipeline

Machine learning pipelines comprise a collection of data, verification, resource management, and DevOps support in the form of large-scale computational resources, all of which are performed in parallel [7]. When working with different kinds of infrastructure, whether cloud-based or on-premise, you must ensure that your models will provide production-ready, accurate data that will eventually change as the infrastructure evolves. The models, on the other hand, are never permanent constructions; rather, they are always altering in response to new data, since model decay necessitates the need to retrain your models over time [7]. It is possible to utilize CI/CD to construct a continuous feedback loop, which ensures that your models are updated and correct without the need for your ongoing attention, supervision, and involvement.

One must start with data while building a machine learning pipeline. It is necessary to do data validation and other data checks to ensure that your information is appropriate [8]. Then there's model training, which involves experimenting with various techniques to discover the best fit for the model. Following this, more model testing is carried out before the deployment of models into production. To determine if model retraining is needed, it is necessary to conduct the deployment and prediction phases safely. After that, one must verify a feedback loop to have the data from the projections checked in data validation. One issue to consider is how often an automated model should be trained [8,9]. It might be as often as every few minutes depending on several variables. This retraining process is handled by continuous integration, or CI, with the understanding that the data may need to meet different rules such as GDPR or other needed limits. One will have more control over your CI/CD pipeline as a result of statistical testing and anomaly detection, which will ensure that your data is legitimate and your models' predictions are correct [9]. Machine learning pipelines that are continuously refined and improved have the distinct advantage of refining and steadily increasing over the years. Maintaining a strong, production-ready pipeline requires regular monitoring and analysis of the accuracy [9]. Using an end-to-end platform such as cnvrg.io reduces the need for considerable DevOps and data science friction, allowing for the use of the tools for an automated CI/CD ML pipeline to be implemented.

E. The Evolution of Continuous Integration/Continuous Delivery and DevOps

Agile ideas and practices have grown in popularity in many software development businesses throughout the 2000s, owing to technological advancements and widespread acceptance [10,11]. As a result of adopting an agile mentality, the amount of time necessary to build new application functionality was reduced, enabling development teams to deploy features often and early. Certain catastrophic consequences, on the other hand, were inevitable since deployment to operations may still take weeks or months. As early as 2010, the emergence of the cloud-enabled businesses and teams to rapidly create their apps and have a working version of the application put into operation in a matter of hours or minutes [11]. This procedure was made regular and low-risk by the introduction of DevOps in enterprises. Organizations are increasingly embracing DevOps ideas and methods to deploy their modifications daily many times. DevOps is at the center of the DevOps movement, as is a continuous integration and continuous delivery (CI/CD) [11].

F. Machine Learning in DevOps

When it comes to DevOps and Machine Learning, there is a significant synergy (ML). The use of machine learning (ML) allows DevOps teams to mine huge complicated datasets, find patterns and antipatterns, reveal new insights, iterate and improve queries, and repeat continually – all at the speed of a computer. Akin to this, machine learning (ML) is in many respects the next generation of automation, expanding on John Willis' and Damon Edwards' prescription for 'CAMS' (computer-aided manufacturing). Because of automation, DevOps allows for a far quicker SDLC, but one that is far too opaque and scattered to be understood by a regular human being, as well as dynamic and ephemeral. However, like with automation, machine learning is ideally suited to dealing with the high velocity, large volume, and wide diversity of data created by new delivery methods and the next wave of composable, atomized, and scaled-out apps [12].

G. How Machine Learning enhances DevOps

i. Keeping track of when an application is delivered

Input from 'DevOps tools' (such as Jira, Git, Jenkins, SonarQube, Puppet, Ansible, and others) offers insight into the delivery process. The use of machine learning (ML) may find abnormalities in that data, such as big code volumes, lengthy build times, sluggish delivery rates, and late code review, to detect many of the "wastes" of application development, such as gold plating, incomplete work, wasteful resourcing, increased work switching, or process downturns.

ii. Assuring the quality of the application

ML can intelligently examine QA results, find new mistakes, and construct a test pattern library based on discoveries by evaluating the output from testing tools. With this machine-driven knowledge of a 'known good release,' we can assure thorough testing of every rollout, even for unique problems, and thereby improve the quality of the apps we supply.

iii. Securing application delivery

The patterns of user activity are as distinct as fingerprints when compared to each other. Adopting ML for detecting abnormalities in Dev and Ops user behavior may aid in the detection of harmful activities. If users access sensitive repositories in an unusual way, such as via automated routines, deployment activities, test execution, or system provisioning, they may be exposing themselves to 'known evil' practices, such as writing back doors, deploying unapproved code, or theft of intellectual property. This may be done purposefully or unintentionally.

iv. Managing the manufacturing process

Due to higher data quantities, user counts, and volume of transactions in production (as opposed to development or testing), machine learning is best used to analyze applications in production. Developer and operations teams may utilize machine learning to identify "typical" patterns such as user volumes, resource usage, and transaction throughput before looking for "abnormal" ones (e.g. DDOS conditions, memory leaks, race conditions, etc.).

v. Managing Alert Storms

ML may be used to manage the large stream of warnings that occur in production systems in a straightforward, practical, and high-value way. Simply using ML alerts tied to groups is one way to do this (e.g. by a common transaction ID; a common set of servers; or a common subnet). Another option is to use a system that learns to distinguish between 'known good' and 'known harmful' warnings over time. Thus, warning storms and alert weariness may be reduced by filtering.

vi. Analytical troubleshooting and triage

This is another area in which modern machine learning (ML) techniques excel. ML is capable of automatically detecting 'known problems,' as well as others that are unknown and can even begin intelligently triaging them. It's possible to spot abnormalities in "normal" processing using machine learning methods, and then associate this problem with a new configuration or deployment using release logs and other information from the logs. Automated tools can notify operations, create ticketing (or chat session), as well as allocate it to the appropriate resource using machine learning [12,13]. A machine-learning algorithm may eventually be able to recommend the optimum solution.

vii. Preventing production failures

When it comes to avoiding system failures, machine learning can go far beyond traditional methods like straight-line capacity planning. An outage may influence customer engagement if ML can map known excellent patterns of use to anticipate how many customers would use a new product, how much infrastructure is required, and so on. Because ML recognizes 'early signs' in systems and apps that would otherwise be invisible, DevOps may begin issue rectification or avoidance much more quickly than they would otherwise be able to [13].

viii. Analyzing business impact

To be successful in DevOps, it is essential to understand the influence of code release on business priorities and objectives. By synthesizing and analyzing real-world user data, machine learning algorithms may identify both positive and negative trends, providing an 'early warning system' to both professionals and business teams when apps are experiencing issues (e.g., by early reports of higher cart turnover or shorter buyer experiences); or are spectacularly productive (e.g. through early detection of high user registrations or click-through rates) [13,14].

III. FUTURE IN THE U.S

The future of DevOps and ML in the U.S is going to focus more on intelligent systems that train on data and learn to complete different functions on their own. Machine Learning Techniques have the potential to make CI/CD Pipeline and DevOps more efficient [14]. It may improve performance by allowing quick development and operation cycles, and by providing a great user experience with these features as well. Data collecting in the DevOps system may be simplified using machine learning algorithms. DevOps is a relatively new concept in the business, and it will grow in importance as artificial intelligence (AI) and data science become more widely used. Using Machine Learning Techniques in DevOps processes offers a viable alternative for companies trying to speed up and enhance their ML solutions, as well as innovate using ML [15,16,17]. As a result, operations like data preparation and model building may be completed more quickly, and procedures can be standardized to enable ML to be used at scale. However, despite the obvious advantages, the operationalization of Machine Learning Techniques is often neglected [17]. The moment has come to prioritize the implementation of artificial intelligence as a strategic goal for the company.

IV. ECONOMIC BENEFITS IN THE UNITED STATES

The United States will benefit from this research by gaining knowledge on machine learning can improve CI/CD and DevOps functionalities. Over the past several years, machine learning use has grown significantly across a wide range of companies and use cases [17,18]. A better consumer experience is enabled by our machine learning algorithms, which also assist avoid safety issues and ensuring market efficiency. More code is released to production faster using automation, as opposed to manual labor and interruptions caused by planned maintenance or other service outages. The use of continuous integration and delivery (CI/CD) helps the development team to be more productive [18]. Organizations that use CI/CD pipelines can provide code more quickly. Because teams can spend more time on enhancing applications and less time on the technical procedures of distributing code to diverse environments, standardizing builds, building tests, and automating deploys may help them do so. CI/CD for FinTech reduces the amount of rework and waiting time. Because regular operations may be automated, software engineers can devote their time and attention to more important activities, such as assuring the quality and security of their code [18]. The U.S capital markets are constantly dealing with investments and transactions, and as a result, they need complete data protection for all of their information. To secure your company's data, a well-protected application must be governed by the highest security standards and regulations available in the Cloud.

V. CONCLUSION

This paper focused on improving CI/CD Pipeline and DevOps Processes by utilizing machine learning techniques. The research aims to give a thorough articulation of CI/CD Pipeline and DevOps problems and associated mitigating solutions. This research utilized a literature review to review past resources on the topic. In applying the findings from this research, stakeholders will be able to avoid issues, manage risks effectively, and budget for the unavoidable obstacles of utilizing CI/CD Pipeline and DevOps. Since its inception, the principle of CI/CD Pipeline and DevOps have helped enterprises move forward by offering cost-effective options for delivering software more quickly while also fostering team cooperation. The advantages of DevOps outweigh the obstacles, though. Security integration is one of the most challenging aspects of the CI/CD Pipeline and DevOps process, which many businesses find tough to implement. However, security must be considered throughout the process. Early and efficient security implementation in CI/CD Pipeline and DevOps may assist in promptly identifying vulnerabilities and resolving operational flaws before they become a serious problem. Embedding ML techniques in the DevOps lifecycle helps to keep it in place and operational throughout the product's lifespan. By using this security measure, the code will be protected against cybersecurity threats and data breaches.

REFERENCES

1. T. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman. Project adam: building an efficient and scalable deep learning training system. In OSDI, 2014.
2. M. Li, D. G. Andersen, J. W. Park, A. J. Smola et al.,"Scaling Distributed Machine Learning with the Parameter Server." OSDI, 2014.
3. T. Kraska, A. Talwalkar, J. C. Duchi, R. Griffith et al., "MLbase: A Distributed Machine-learning System," CIDR, 2013. [12]Scikit-Learn; Licence:New BSD License; Available online link:„https://scikit-learn.org/stable/"; Original author: David Cournapeau; stable release data: 0.22 / 3 December 2019.
4. M. Akbar, S. Mahmood, M. Shafiq, A. Alsanad, A. Alsanad and A. Gumaei, "Identification and prioritization of DevOps success factors using fuzzy-AHP approach", Soft Computing, 2020.
5. T. Chaghrouchni, I. Kabbaj and Z. Bakkoury, "Machine Learning in Predicting the Appropriate Model of Software Process-Models Deviation", Journal of Digital Information Management, vol. 16, no. 6, p. 308, 2018.
6. A. Mackworth, "Model-Driven Interpretation in Intelligent Vision Systems", Perception, vol. 5, no. 3, pp. 349-370, 1976.
7. S. Maggo and C. Gupta, "A Machine Learning based Efficient Software Reusability Prediction Model for Java Based Object Oriented Software", International Journal of Information Technology and Computer Science, vol. 6, no. 2, pp. 1-13, 2014.
8. R. Malhotra, "A systematic review of machine learning techniques for software fault prediction", Applied Soft Computing, vol. 27, pp. 504-518, 2015.
9. Q. Song, X. Zhu, G. Wang, H. Sun, H. Jiang, C. Xue, B. Xu and W. Song, "A machine learning based software process model recommendation method", Journal of Systems and Software, vol. 118, pp. 85-100, 2016.
10. T. Al-Shehari and F. Shahzad, "Improving Operating System Fingerprinting using Machine Learning Techniques", International Journal of Computer Theory and Engineering, pp. 57-62, 2014.
11. D. Ceballos, D. López-Álvarez, G. Isaza, R. Tabares-Soto, S. Orozco-Arias and C. Ferrin, "A Machine Learning-based Pipeline for the Classification of CTX-M in Metagenomics Samples", Processes, vol. 7, no. 4, p. 235, 2019.
12. A. Costea, "Applying Fuzzy Logic and Machine Learning Techniques in Financial Performance Predictions", Procedia Economics and Finance, vol. 10, pp. 4-9, 2014.
13. D. Faisal, "An Integrated Framework to Automate the Prediction of oil by Applying Machine Learning Techniques on the Information Retrieved from Upstream Segment", Journal of Advanced Research in Dynamical and Control Systems, vol. 12, no. 4, pp. 320-331, 2020.
14. P. Jayaraman and R. Parthasarathi, "Performance counter based online pipeline bugs detection using machine learning techniques", Microprocessors and Microsystems, vol. 84, p. 104262, 2021.
15. Y. Kakde and S. Agrawal, "Predicting Survival on Titanic by Applying Exploratory Data Analytics and Machine Learning Techniques", International Journal of Computer Applications, vol. 179, no. 44, pp. 32-38, 2018.
16. Karamitsos, S. Albarhami and C. Apostolopoulos, "Applying DevOps Practices of Continuous Automation for Machine Learning", Information, vol. 11, no. 7, p. 363, 2020.
17. A. Kjamilji, "Techniques and Challenges while Applying Machine Learning Algorithms in Privacy Preserving Fashion", Proceeding International Conference on Science and Engineering, vol. 3, p. xix-xix, 2020.
18. L. Sachin, "Automated Performance Indicator System for CI/CD & DevOps Developers of Software Industry using Python", International Journal for Research in Applied Science and Engineering Technology, vol. 8, no. 7, pp. 373-379, 2020.