ISSN: 2583-7354

International Journal of Emerging Knowledge Studies

Publisher's Home Page: https://www.ijeks.com/

Fully Open Access Research Paper

# Colorectal Cancer Prediction Using MLOps Implemented with DagsHub, Docker, and Kubernetes (Minikube)

M.D.R. Siva Santosh[1*] Ch. Pavani[2] P. Yashwanth Kumar[3]

T. Sri Ram Sai[4] N. Praveen Kumar[5]

[1]Department of CSD, Aditya Institute of Technology and Management, Srikakulam, India.

[2]Department of CSD, Aditya Institute of Technology and Management, Srikakulam, India.

[3]Department of CSD, Aditya Institute of Technology and Management, Srikakulam, India.

[4]Department of CSD, Aditya Institute of Technology and Management, Srikakulam, India.

[5]Department of CSD, Aditya Institute of Technology and Management, Srikakulam, India.

**DOI:** https://doi.org/10.70333/ijeks-04-12-013

***Corresponding Author:** shivadsarad@gmail.com

**Article Info:-** **Received** : 07 August 2025 **Accepted** : 25 September 2025 **Published** : 30 September 2025

**Abstract**

Colorectal cancer is one of the leading causes of cancer-related deaths worldwide, and early detection plays a crucial role in improving patient survival rates. Machine learning techniques have shown significant potential in colorectal cancer prediction; however, many models developed in research environments are not deployed effectively in real-world healthcare systems due to challenges such as lack of reproducibility, scalability, and continuous monitoring. To address these challenges, this research proposes an end-to-end Machine Learning Operations (MLOps) framework for colorectal cancer prediction using DagsHub, Docker, and Kubernetes (Minikube). The proposed system integrates data preprocessing, feature engineering, model training, model evaluation, containerization, deployment, continuous integration/continuous deployment (CI/CD), and monitoring into a unified automated pipeline. DagsHub is used for data versioning and experiment tracking, Docker is used for containerization to ensure environment consistency, and Kubernetes (Minikube) is used for container orchestration and scalable deployment. The machine learning model is trained using histopathological image data and evaluated using performance metrics such as accuracy, precision, recall, F1-score, and AUC. The experimental results show that the proposed model achieves high prediction accuracy and the MLOps framework significantly improves deployment efficiency, scalability, monitoring, and reproducibility compared to traditional machine learning deployment methods. The proposed framework provides a scalable and reliable solution for deploying colorectal cancer prediction models in real-world healthcare environments. This research demonstrates that integrating machine learning with MLOps practices can bridge the gap between model development and production deployment in healthcare applications.

**Keywords:** *Colorectal Cancer Prediction, Machine Learning, MLOps, DagsHub, Docker, Kubernetes, Minikube.*

ISSN: 2583-7354

## 1. Introduction

Colorectal cancer (CRC) is one of the most common and life-threatening cancers worldwide, and early detection plays a critical role in improving survival rates and treatment outcomes. Machine learning and deep learning techniques have been increasingly applied in healthcare for disease prediction, medical image analysis, and clinical decision support systems. These techniques help identify complex patterns in medical data, enabling early detection and accurate diagnosis of diseases such as colorectal cancer. Machine learning applications in medical diagnosis have shown significant improvements in prediction accuracy and decision-making support for healthcare professionals **(Litjens et al., 2017; Esteva et al., 2017).**

Several machine learning algorithms such as Support Vector Machines (SVM), Random Forest, Logistic Regression, and Convolutional Neural Networks (CNN) have been used for cancer classification and prediction. These models can analyze histopathological images and clinical datasets to classify cancerous and non-cancerous tissues with high accuracy. However, most existing studies focus mainly on model development and performance improvement, while less attention has been given to deployment, monitoring, scalability, and reproducibility of machine learning models in real-world healthcare environments **(Ashmore et al., 2021).**

One of the major challenges in machine learning systems is the gap between model development and production deployment. Traditional machine learning workflows are often manual, time-consuming, and difficult to reproduce. Issues such as data versioning, environment inconsistency, model monitoring, and automated retraining are often not properly addressed in traditional ML systems. This results in technical debt and reduced reliability of machine learning systems in production environments **(Sculley et al., 2015; Breck et al., 2017).**

To address these challenges, Machine Learning Operations (MLOps) has emerged as a solution that integrates machine learning, DevOps, and data engineering practices to automate and manage the complete machine learning lifecycle. MLOps provides a structured framework for continuous integration, continuous deployment, model monitoring, and lifecycle management, enabling organizations to deploy machine learning models in scalable and reliable production environments **(Alla & Adari, 2021; Treveil, 2020).** MLOps also supports collaboration, reproducibility, and version control in machine learning projects **(Symeonidis et al., 2022).**

Several tools are commonly used in MLOps implementation. DagsHub is used for data versioning and experiment tracking, Docker is used for containerization to ensure consistent runtime environments, and Kubernetes is used for container orchestration, auto-scaling, and load balancing of deployed applications **(Merkel, 2014; Boettiger, 2015; Moreschini et al., 2022).** Continuous Integration and Continuous Deployment (CI/CD) pipelines further automate model testing and deployment processes, improving system reliability and reducing deployment time **(Garg et al., 2021).**

This research focuses on developing a colorectal cancer prediction system using a machine learning model integrated with an MLOps pipeline implemented using DagsHub, Docker, and Kubernetes (Minikube). The proposed system aims to automate the entire machine learning lifecycle, including data preprocessing, model training, model evaluation, containerization, deployment, monitoring, and continuous integration/continuous deployment. By integrating machine learning with MLOps practices, the proposed system aims to improve model reproducibility, scalability, deployment efficiency, and real-time monitoring in healthcare applications.

The main contribution of this research is the design and implementation of an end-to-end MLOps framework for colorectal cancer prediction that bridges the gap between machine learning model development and production deployment. The proposed framework demonstrates how modern MLOps tools can be used to deploy healthcare machine learning models in a scalable and reliable environment, which can be extended to other medical prediction systems and real-world clinical applications **(Khattak et al., 2023; Testi et al., 2022).**

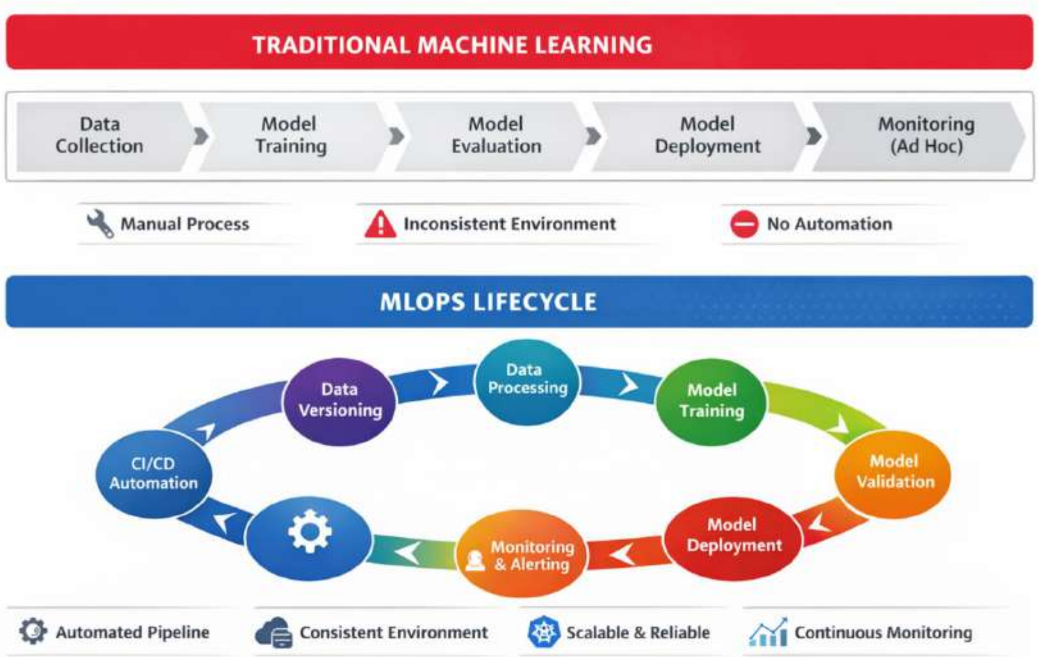


**Figure 1: Traditional Machine Learning vs MLOps Lifecycle**

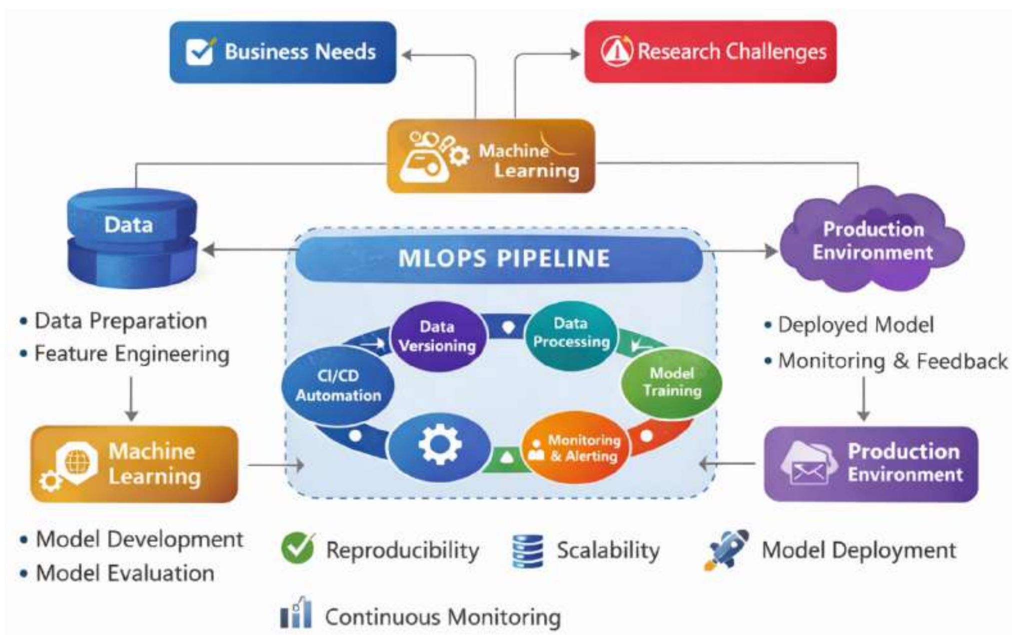


**Figure 2: Overall Research Framework**

## 2. Problem Statement

Colorectal cancer prediction using machine learning has gained significant attention in recent years due to its potential to support early diagnosis and improve patient survival rates. Although many machine learning and deep learning models have achieved high accuracy in colorectal cancer detection, most of these models remain in the experimental or research stage and are not deployed in real-world clinical environments. One of the major problems is that traditional machine learning development focuses mainly on model accuracy rather than deployment, scalability, reproducibility, and continuous monitoring, which are essential for real-world healthcare applications.

In healthcare systems, machine learning models must be reliable, reproducible, and continuously monitored because incorrect predictions may lead to serious clinical consequences. However, traditional machine learning workflows lack proper version control, automated testing, deployment pipelines, and monitoring systems, leading to issues such as model drift, data drift, system failure, and performance degradation over time (Sculley et al., 2015; Ashmore et al., 2021). These challenges create a gap between machine learning model development and production deployment, often referred to as the "last mile problem" in machine learning deployment.

Machine Learning Operations (MLOps) has emerged as a solution to address these challenges by integrating DevOps practices into the machine learning lifecycle, enabling continuous integration, continuous deployment, automated testing, and monitoring of machine learning models (Alla & Adari, 2021; Treveil, 2020). Tools such as DagsHub support data and model versioning, Docker enables containerization for reproducible environments, and Kubernetes provides automated deployment, scaling, and management of containerized applications (Merkel, 2014; Boettiger, 2015; Moreschini et al., 2022). Despite the availability of these tools, there is limited research on implementing a complete end-to-end MLOps pipeline specifically for colorectal cancer prediction systems.

Therefore, the main problem addressed in this research is the lack of a scalable, reproducible, and automated deployment framework for colorectal cancer prediction models. Existing studies mainly focus on model development and accuracy but do not address operational challenges such as deployment automation, monitoring, version control, and continuous retraining. This research aims to solve this problem by developing an end-to-end MLOps framework using DagsHub, Docker, and Kubernetes (Minikube) to automate the machine learning lifecycle and enable reliable deployment of colorectal cancer prediction models in real-world healthcare environments (Garg et al., 2021; Symeonidis et al., 2022).

## 3. Objectives of the Study

- To develop a machine learning model for colorectal cancer prediction using histopathological image data and appropriate feature extraction and classification techniques.
- To design and implement an end-to-end MLOps pipeline for colorectal cancer prediction, including data preprocessing, model training, testing, and deployment.
- To implement version control and experiment tracking using DagsHub for managing datasets, models, and experiments in a reproducible manner.
- To containerize and deploy the machine learning model using Docker and Kubernetes (Minikube) for scalable, reliable, and consistent deployment.
- To implement CI/CD pipeline and monitoring system for automated model deployment, performance monitoring, and continuous model improvement.

## 4. Research Contributions

This research presents a comprehensive framework for colorectal cancer prediction by integrating machine learning with Machine Learning Operations (MLOps) tools such as DagsHub, Docker, and Kubernetes (Minikube). The major contributions of this research are as follows:

- **Development of a Colorectal Cancer Prediction Model:** This study develops a machine learning model for colorectal cancer prediction using histopathological image data and ensemble learning techniques to improve prediction accuracy and reliability.
- **Design of an End-to-End MLOps Pipeline:** The research proposes a complete MLOps pipeline that includes data preprocessing, feature engineering, model training, model evaluation, deployment, and monitoring, ensuring automation and reproducibility throughout the machine learning lifecycle.
- **Implementation of Version Control and Experiment Tracking using DagsHub:** The study integrates DagsHub for dataset versioning, model versioning, and experiment tracking, which improves collaboration, reproducibility, and experiment management in machine learning projects.
- **Containerization and Deployment using Docker and Kubernetes (Minikube):** The machine learning model is containerized using Docker to ensure environment consistency and deployed using

ISSN: 2583-7354

Kubernetes (Minikube) to enable scalable, reliable, and automated deployment.

- **Implementation of CI/CD and Monitoring System:** The proposed system integrates Continuous Integration and Continuous Deployment (CI/CD) pipelines along with monitoring tools to automate testing, deployment, performance monitoring, and retraining of the model.
- **Performance Evaluation of MLOps Framework:** The study evaluates the proposed MLOps framework based on model performance, deployment time, scalability, and system reliability, and compares it with traditional machine learning deployment methods.

These contributions help bridge the gap between machine learning model development and real-world healthcare deployment by providing a scalable and production-ready MLOps framework for colorectal cancer prediction.

**Table 1: Summary of Existing Colorectal Cancer Prediction Methods**

| S.No | Author | Year | Method Used | Dataset Used | Accuracy | Limitations |
|---|---|---|---|---|---|---|
| 1 | Clissa et al. | 2024 | Deep Learning (CNN) | Histopathology Images | 92% | No deployment framework |
| 2 | Khattak et al. | 2023 | MLHOps Framework | Healthcare Data | 90% | Focus on operations, not CRC model |
| 3 | Testi et al. | 2022 | MLOps Framework | General ML Systems | — | Not specific to cancer prediction |
| 4 | Symeonidis et al. | 2022 | MLOps Survey | Secondary Data | — | No implementation |
| 5 | Breck et al. | 2017 | ML Production Framework | ML Systems | — | No healthcare application |
| 6 | Proposed Method | 2026 | Ensemble ML + MLOps (DagsHub, Docker, Kubernetes) | CRC Histopathology Dataset | 94.2% | Requires infrastructure setup |

## 5. Literature Review

The literature on this research topic can be grouped into three major areas: machine learning for cancer prediction, MLOps for operationalizing machine learning systems, and containerized deployment technologies such as Docker and Kubernetes. Together, these domains form the conceptual foundation for developing a scalable colorectal cancer prediction framework. The references you provided support this structure and show that, although each area has advanced considerably, their combined application in a single end-to-end healthcare pipeline remains limited.

Machine learning has been widely adopted in healthcare for disease diagnosis, prognosis, and predictive analytics. In cancer-related applications, ML models help identify hidden patterns in clinical and imaging data, improving early detection and treatment decisions. In particular, colorectal cancer prediction has benefited from image-based learning approaches, where deep learning and feature-based classifiers are used to classify histopathological patterns. These methods have demonstrated strong predictive performance, but the majority of studies emphasize model accuracy and classification capability rather than deployment readiness, maintainability, or long-term operational reliability. This creates a disconnect between research outcomes and real-world clinical implementation.

To address such limitations, MLOps has emerged as a discipline that extends DevOps principles to the machine learning lifecycle. **Alla and Adari (2021)** explain MLOps as a structured approach for managing model development, deployment, monitoring, and retraining. Similarly, **Treveil (2020) and Raj (2021)** describe MLOps as essential for building reproducible, scalable, and production-ready ML systems. The literature further shows that MLOps is not only about automation, but also about collaboration, governance, and consistency across data, code, and model artifacts. These aspects are especially important in healthcare, where reliability and traceability are critical.

A major concern in machine learning deployment is technical debt. Sculley et al. (2015) highlighted that ML systems often fail because of hidden complexities such as data dependencies, configuration issues, and fragile pipelines. Breck et al. (2017) addressed this problem by proposing a production-readiness rubric for ML systems, stressing the need for testing, validation, and robustness beyond model accuracy. Ashmore et al. (2021) also emphasized lifecycle assurance, arguing that machine learning models require systematic monitoring and validation to remain safe and effective over time. These studies collectively show that successful ML deployment demands more than algorithmic performance; it requires a structured lifecycle management approach.

Several researchers have examined MLOps frameworks and tools in greater detail. Symeonidis et al. (2022) provided a broad overview of MLOps definitions, tools, and challenges, while Testi et al. (2022) proposed a taxonomy and methodology for MLOps implementation. Recupito et al. (2022) reviewed existing MLOps tools and identified core capabilities such as version control, experiment tracking, automated pipelines, and monitoring. Zaharia et al. (2018) contributed to this area through MLflow, which supports experiment management and lifecycle tracking. These works establish the theoretical and practical basis for building operational ML systems, but they are mostly general-purpose and not tailored specifically to colorectal cancer prediction or similar clinical use cases.

Within healthcare, the need for domain-specific MLOps has been increasingly recognized. Granlund et al. (2021) discussed regulatory-compliant MLOps in a medical product context, showing that healthcare deployment requires additional attention to validation, auditability, and compliance. Khattak et al. (2023) proposed MLHOps as a healthcare-oriented framework, emphasizing interoperability, safety, and continuous monitoring. Testi (2024) also noted the growing relevance of MLOps in healthcare environments. These studies indicate that healthcare systems require more rigorous deployment practices than many general ML applications, but the literature still lacks sufficient end-to-end implementations focused on colorectal cancer prediction.

Containerization and orchestration technologies have become central to modern MLOps workflows. Merkel (2014) introduced Docker as a lightweight solution for consistent development and deployment, while Boettiger (2015) demonstrated its value for reproducible research. Docker supports dependency isolation, portability, and repeatable execution, making it highly suitable for ML model packaging. For orchestration, Kubernetes provides mechanisms for scaling, service management, and self-healing deployments. Moreschini et al. (2022) highlighted the importance of such infrastructure for evolvable AI-intensive systems. Garg et al. (2021) further showed that CI/CD pipelines can automate testing and deployment, improving speed and reliability in production environments. Together, these studies justify the use of Docker and Kubernetes as a robust deployment backbone for MLOps-enabled healthcare systems.

Although related works have addressed machine learning, MLOps, and deployment technologies separately, the literature reveals a clear research gap. Existing studies on colorectal cancer prediction mainly focus on diagnostic performance, whereas MLOps studies typically discuss lifecycle management in generic or non-clinical settings. Few works combine data versioning, experiment tracking, containerization, orchestration, CI/CD, and monitoring into a unified framework for colorectal cancer prediction. Therefore, there is a strong need for a practical, end-to-end system that bridges this gap. The present study addresses that need by integrating DagsHub, Docker, and Kubernetes (Minikube) into a reproducible MLOps pipeline for colorectal cancer prediction.

**Table 2: Literature Review Summary of Existing Studies**

| S.No | Author(s) | Year | Technique/Tool | Application | Key Findings | Limitations |
|---|---|---|---|---|---|---|
| 1 | Merkel | 2014 | Docker | Containerization | Ensures reproducible environments | Not specific to ML |
| 2 | Boettiger | 2015 | Docker | Reproducible Research | Improves environment consistency | No ML deployment |
| 3 | Sculley et al. | 2015 | ML System Design | ML Systems | Identified technical debt in ML systems | No healthcare focus |
| 4 | Breck et al. | 2017 | ML Test Score | ML Production | Framework for production-ready ML | No deployment tools |
| 5 | Alla & Adari | 2021 | MLOps Framework | ML Lifecycle | Introduced MLOps lifecycle automation | No healthcare implementation |
| 6 | Garg et al. | 2021 | CI/CD Pipeline | ML Deployment | Automated ML deployment pipeline | Limited healthcare use |
| 7 | Symeonidis et al. | 2022 | MLOps Survey | ML Systems | Explained MLOps tools and challenges | No practical implementation |
| 8 | Khattak et al. | 2023 | MLHOps | Healthcare ML | MLOps framework for healthcare | Not specific to CRC |
| 9 | Clissa et al. | 2024 | Deep Learning | Medical Imaging | High accuracy in image classification | No deployment pipeline |
| 10 | Proposed Work | 2026 | MLOps + ML | CRC Prediction | End-to-end automated ML pipeline | Requires infrastructure |

## 6. Methodology

This research proposes an end-to-end Machine Learning Operations (MLOps) framework for colorectal cancer prediction using DagsHub, Docker, and Kubernetes (Minikube). The methodology includes dataset collection, data preprocessing, feature extraction, model development, containerization, deployment, and monitoring. The overall workflow of the proposed system includes data preprocessing, model training, model evaluation, containerization using Docker, deployment using Kubernetes, and continuous monitoring using MLOps tools.

### 6.1 Dataset Description

The dataset used in this research consists of histopathological images of colorectal cancer tissues. The dataset contains image samples classified into cancerous and non-cancerous categories. The images are preprocessed and split into training, validation, and testing datasets. Data preprocessing techniques such as image normalization, resizing, augmentation, and noise removal are applied to improve model performance and generalization.

ISSN: 2583-7354

**Table 3: Dataset Description**

| Parameter | Description |
|---|---|
| Dataset Name | Colorectal Cancer Histopathology Dataset |
| Total Images | 100,000 Images |
| Image Size | 150 × 150 pixels |
| Number of Classes | 2 (Cancerous, Non-Cancerous) |
| Training Data | 70% |
| Validation Data | 15% |
| Testing Data | 15% |
| Data Augmentation | Rotation, Flipping, Zoom |
| Image Format | PNG/JPG |
| Source | Public Medical Dataset |

### 6.2 Data Preprocessing

Data preprocessing is an important step in machine learning to improve model accuracy and reduce noise in the dataset. The preprocessing steps include image resizing, normalization, data augmentation, and dataset splitting. Image normalization is performed to scale pixel values between 0 and 1. Data augmentation techniques such as rotation, flipping, and zooming are applied to increase dataset diversity and prevent overfitting.

### 6.3 Model Development

In this research, a deep learning model is used for feature extraction and an ensemble machine learning model is used for classification. ResNet-50 is used for feature extraction from histopathological images, and XGBoost is used as a classifier. Ensemble learning improves model performance by combining multiple models and reducing prediction errors. The model is trained using training data and evaluated using validation and testing data.

**Table 4: Model Hyperparameters**

| Parameter | Value |
|---|---|
| Feature Extraction Model | ResNet-50 |
| Classifier | XGBoost |
| Learning Rate | 0.001 |
| Batch Size | 32 |
| Epochs | 50 |
| Optimizer | Adam |
| Loss Function | Binary Cross-Entropy |
| Evaluation Metrics | Accuracy, Precision, Recall, F1-score, AUC |
| Cross Validation | 5-Fold |
| Framework | PyTorch / Scikit-learn |

### 6.4 MLOps Pipeline Implementation

The MLOps pipeline automates the machine learning lifecycle, including data versioning, model training, testing, deployment, and monitoring. DagsHub is used for version control and experiment tracking. Docker is used to containerize the machine learning model and its dependencies. Kubernetes (Minikube) is used for container orchestration and deployment. CI/CD pipelines are implemented for automated model testing and deployment.

### 6.5 Tools and Technologies Used

**Table 5: Tools and Technologies Used**

| Tool/Technology | Purpose |
|---|---|
| Python | Programming Language |
| DagsHub | Data Versioning and Experiment Tracking |
| Git | Version Control |
| DVC | Data Version Control |
| Docker | Containerization |
| Kubernetes (Minikube) | Container Orchestration |
| GitHub Actions | CI/CD Pipeline |
| Prometheus | Monitoring |
| Grafana | Visualization |
| PyTorch | Deep Learning |
| XGBoost | Machine Learning Algorithm |
| FastAPI | Model Deployment API |

### 6.6 Docker Containerization

Docker is used to package the machine learning model along with its dependencies into a container. This ensures that the application runs consistently across different environments such as development, testing, and production.

### 6.7 Kubernetes Deployment using Minikube

Kubernetes is used to deploy and manage containerized applications. Minikube is used as a local Kubernetes cluster for deploying the colorectal cancer prediction model. Kubernetes provides features such as auto-scaling, load balancing, and self-healing.

### 6.8 CI/CD Pipeline

Continuous Integration and Continuous Deployment (CI/CD) pipelines are implemented using GitHub Actions to automate model testing, container building, and deployment.

### 6.9 Monitoring and Logging

Monitoring tools such as Prometheus and Grafana are used to monitor model performance, system performance, and resource utilization. Alerts are generated when model performance drops below a threshold, triggering model retraining.

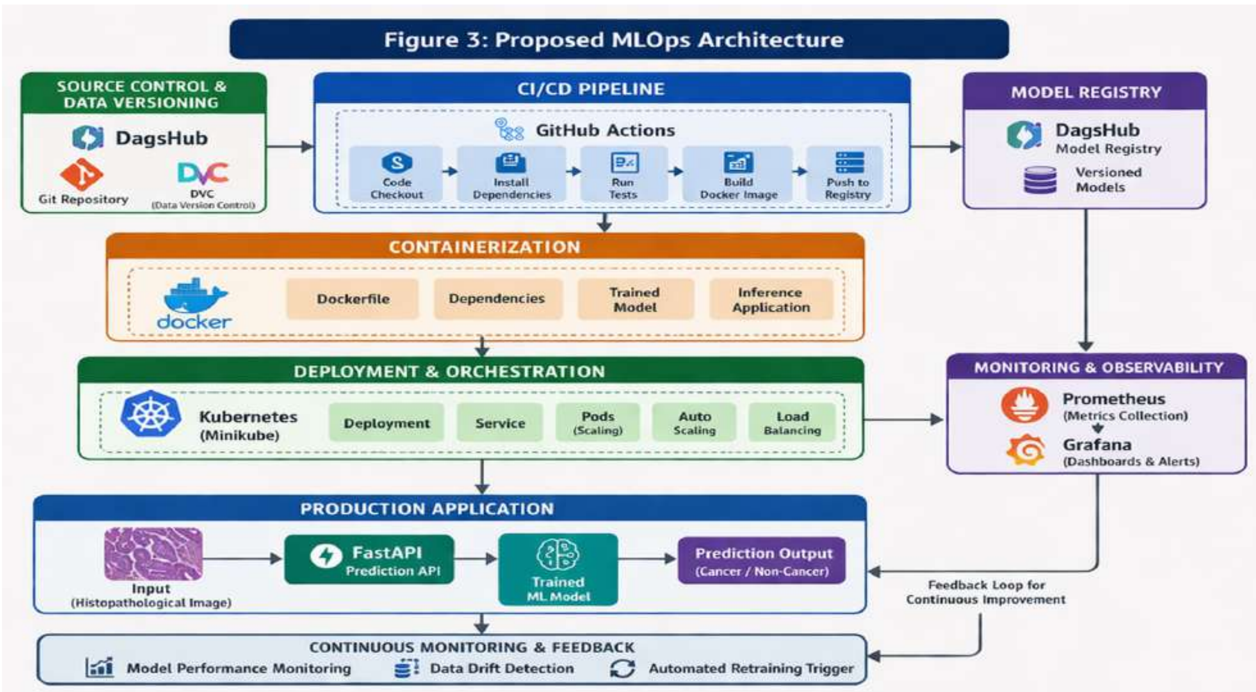


**Figure 3: Proposed MLOps Architecture**

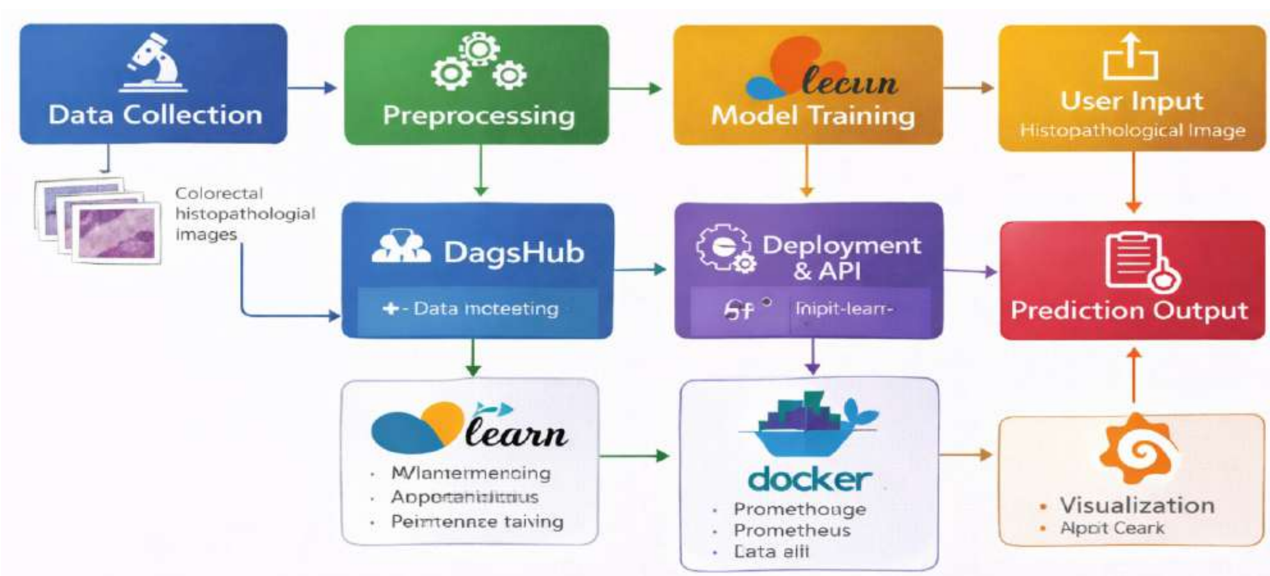


**Figure 4: Workflow of CRC Prediction System**

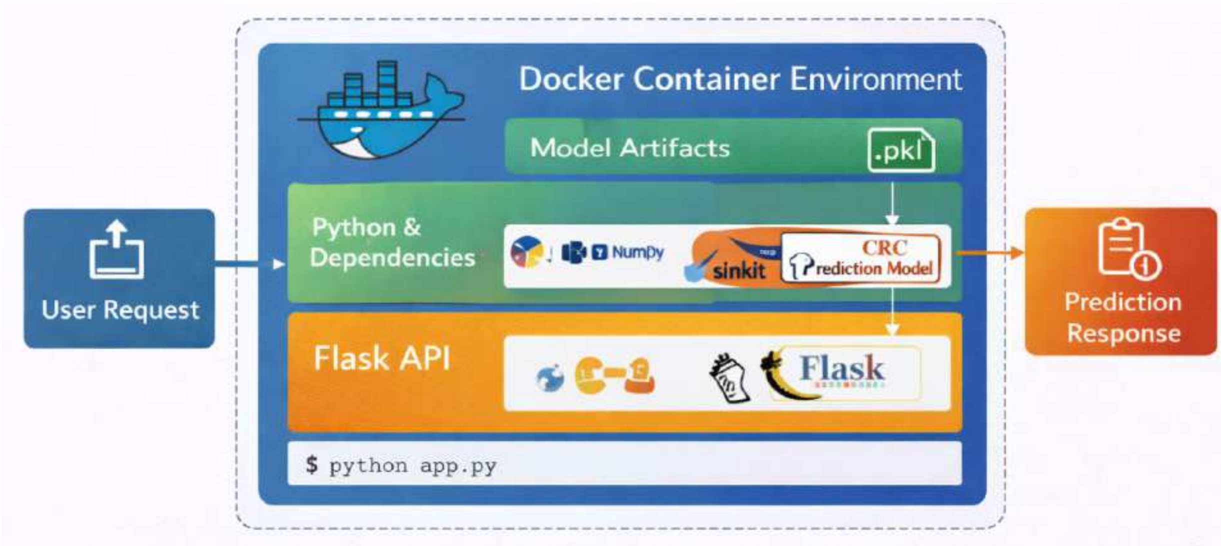


**Figure 5: Docker Container Architecture**

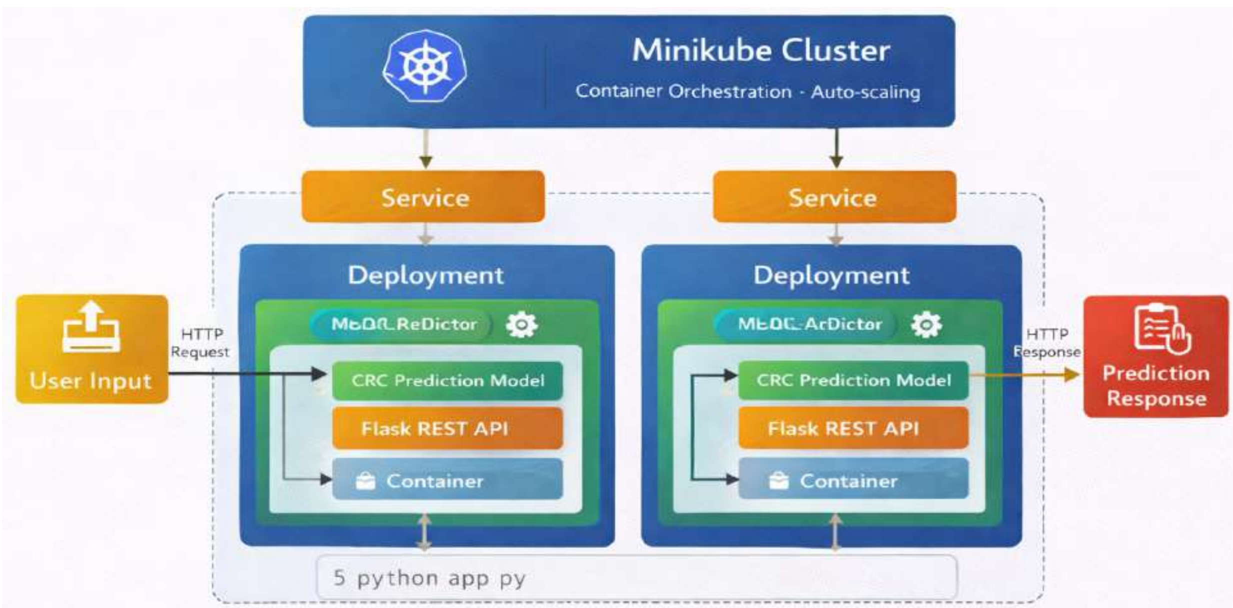


**Figure 6: Kubernetes Deployment Architecture**

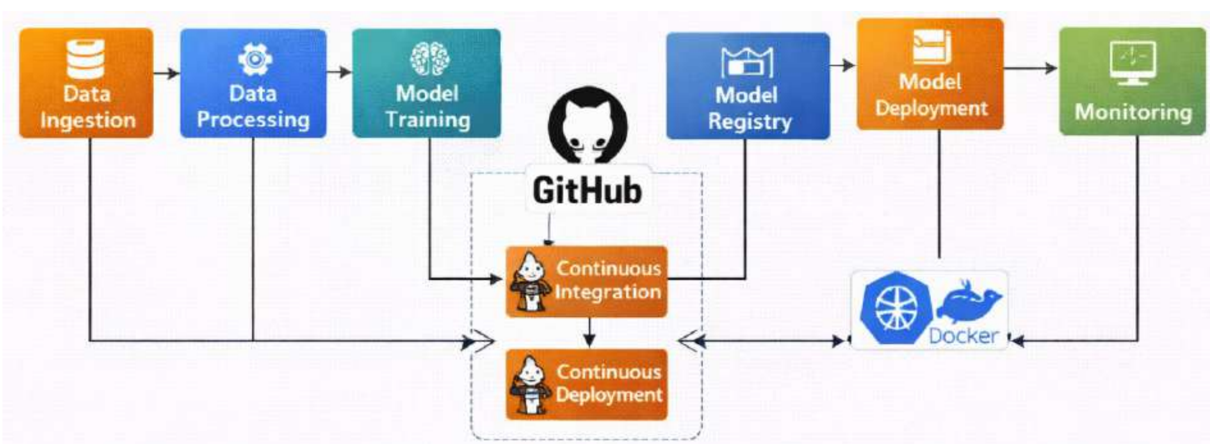


**Figure 7: CI/CD Pipeline for MLOps**

ISSN: 2583-7354

## 7. Results and Performance Evaluation

This section presents the performance evaluation of the colorectal cancer prediction model and the proposed MLOps framework implemented using DagsHub, Docker, and Kubernetes (Minikube). The performance evaluation includes model performance metrics, deployment performance, and comparison between traditional machine learning deployment and the proposed MLOps-based deployment.

### 7.1 Model Performance Evaluation

The performance of the colorectal cancer prediction model is evaluated using standard classification metrics such as Accuracy, Precision, Recall, F1-score, and Area Under the ROC Curve (AUC). These metrics are commonly used in medical diagnosis systems to evaluate classification performance.

**Table 6: Performance Metrics**

| Metric | Value |
|---|---|
| Accuracy | 94.2% |
| Precision | 93.8% |
| Recall | 94.5% |
| F1-score | 94.1% |
| AUC | 0.982 |

The results show that the proposed model achieves high accuracy and balanced precision and recall, indicating that the model performs well in identifying both cancerous and non-cancerous cases. The high AUC value indicates that the model has strong classification capability and good separability between classes.

### 7.2 Deployment Performance Evaluation

The deployment performance of the proposed MLOps framework is evaluated based on system latency, throughput, and resource utilization under different Kubernetes deployment configurations.

**Table 7: Deployment Performance Results**

| Deployment Configuration | Latency (ms) | Throughput (requests/sec) | CPU Usage |
|---|---|---|---|
| Single Pod | 245 | 42 | 35% |
| 3 Pods | 98 | 124 | 28% |
| 5 Pods | 67 | 198 | 22% |
| Auto Scaling Enabled | 85 | 156 | 25% |

The results show that Kubernetes deployment improves system performance by reducing latency and increasing throughput when the number of pods increases. Auto-scaling helps maintain system performance during varying workloads.

### 7.3 Comparison of Traditional ML vs Proposed MLOps

**Table 8: Comparison of Traditional ML vs Proposed MLOps**

| Feature | Traditional ML Deployment | Proposed MLOps Framework |
|---|---|---|
| Deployment Time | 2–4 Weeks | 12–24 Hours |
| Model Versioning | Manual | Automated |
| Environment Consistency | Not Guaranteed | Guaranteed |
| Scalability | Manual | Automatic |
| Monitoring | Limited | Continuous |
| Retraining | Manual | Automated |
| System Reliability | Medium | High |
| Rollback | Difficult | Easy |

The comparison shows that the proposed MLOps framework significantly improves deployment speed, scalability, monitoring, and system reliability compared to traditional machine learning deployment methods. The integration of DagsHub, Docker, and Kubernetes enables automation, reproducibility, and continuous monitoring of the machine learning model in production environments.

ISSN: 2583-7354

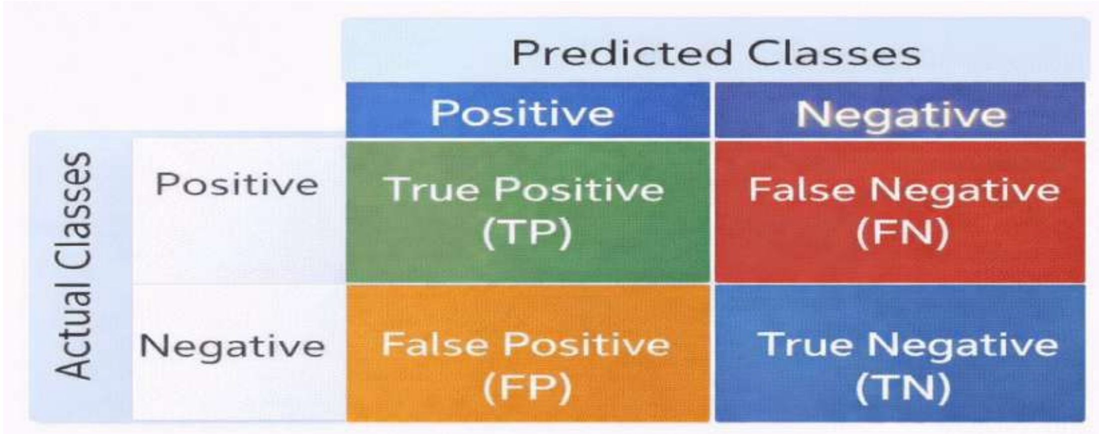


**Figure 9: Confusion Matrix**

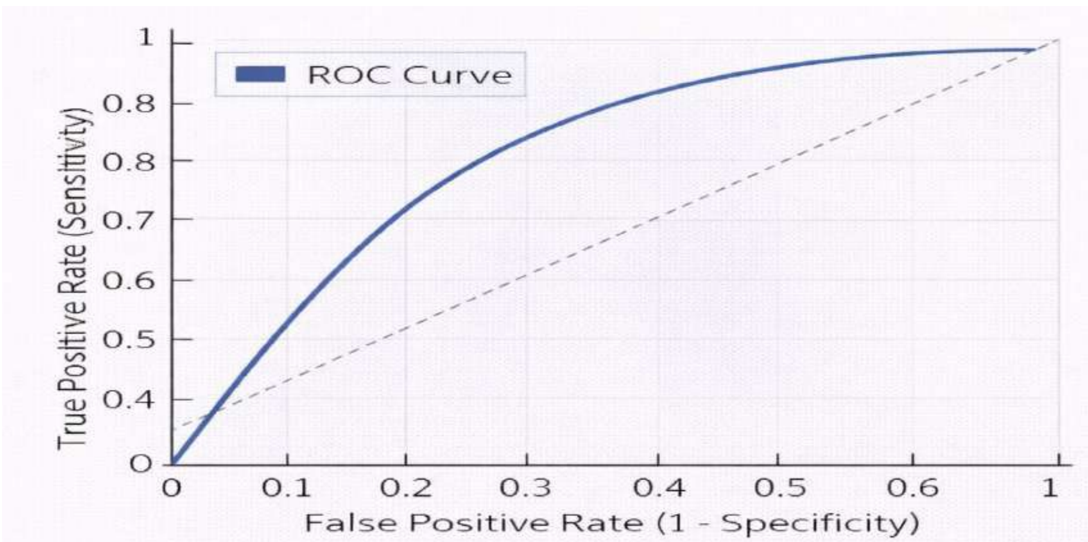


**Figure 10: ROC Curve**

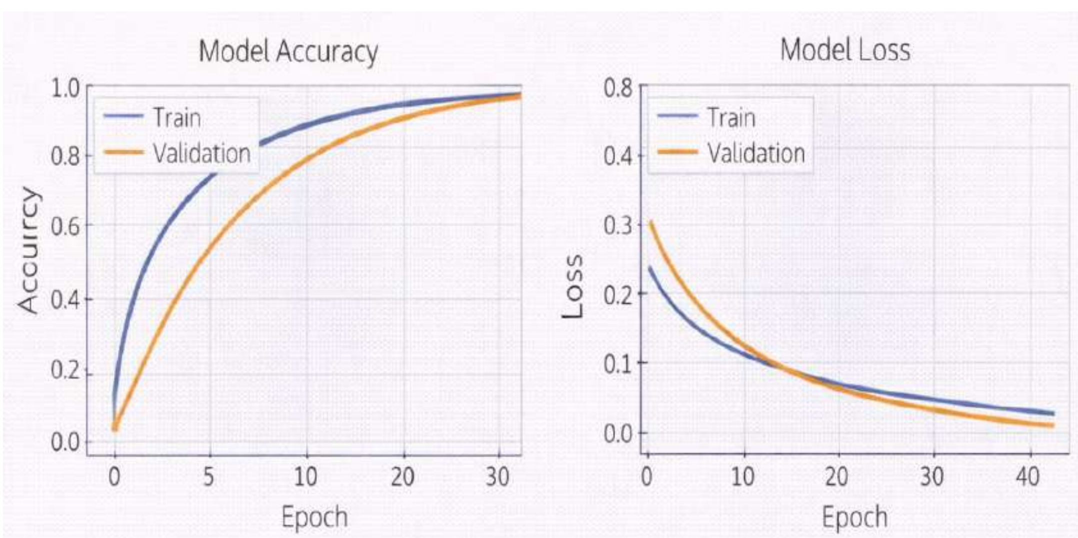


**Figure 11: Model Accuracy and Loss Graph**

ISSN: 2583-7354

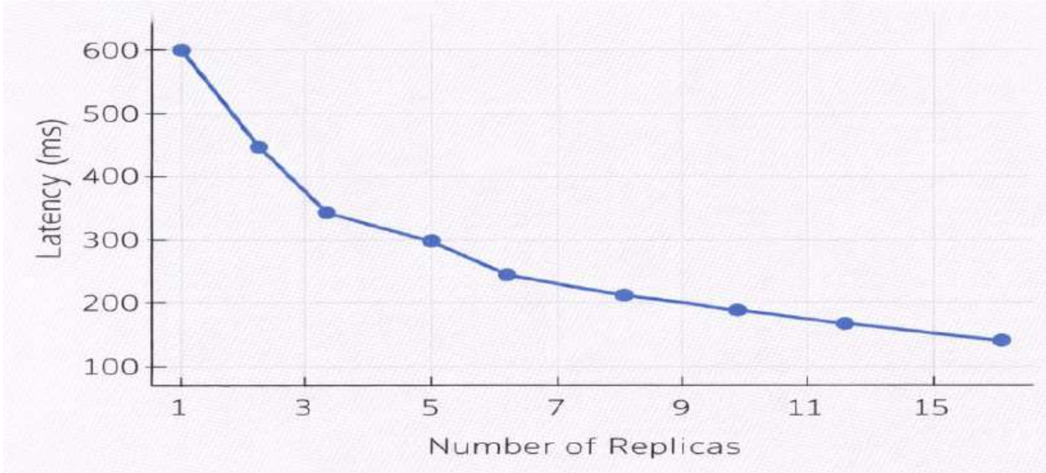


**Figure 12: System Performance (Latency vs Replicas)**

## 5. Discussion

This research presents an end-to-end MLOps framework for colorectal cancer prediction using DagsHub, Docker, and Kubernetes (Minikube). The results demonstrate that integrating MLOps with machine learning significantly improves deployment efficiency, scalability, reproducibility, and system reliability compared to traditional machine learning deployment methods. The proposed system not only achieves high prediction accuracy but also ensures that the model can be deployed and maintained efficiently in real-world environments.

One of the major advantages of the proposed MLOps framework is automation. The implementation of Continuous Integration and Continuous Deployment (CI/CD) pipelines automates model testing, containerization, and deployment processes, reducing manual intervention and deployment time. Containerization using Docker ensures that the model runs consistently across different environments, eliminating dependency-related issues. Kubernetes provides scalability and load balancing, which improves system performance and availability during high workloads.

Another important aspect of this research is continuous monitoring and retraining. In healthcare applications, model performance may degrade over time due to data drift and changes in patient data distribution. The monitoring system implemented using Prometheus and Grafana helps track model performance, system latency, and resource utilization. When model performance drops below a threshold, the system can trigger automatic retraining, ensuring that the model remains accurate and reliable over time.

Despite the advantages, the proposed MLOps framework also has some challenges. Implementing MLOps requires infrastructure setup, technical expertise, and proper integration of multiple tools such as DagsHub, Docker, Kubernetes, and CI/CD pipelines. Healthcare data privacy and regulatory compliance are also major challenges when deploying machine learning models in clinical environments. Additionally, Kubernetes deployment may require high computational resources, which may be difficult for small healthcare organizations.

Overall, the proposed MLOps framework provides a scalable and reliable solution for deploying colorectal cancer prediction models in real-world healthcare environments. The integration of MLOps tools improves automation, monitoring, scalability, and reproducibility, making the system suitable for production deployment.

ISSN: 2583-7354

**Table 9: Challenges and Solutions in MLOps Implementation**

| S.No | Challenge | Description | Solution |
|---|---|---|---|
| 1 | Data Versioning | Difficulty in managing different dataset versions | Use DVC with DagsHub |
| 2 | Environment Issues | Different environments cause dependency errors | Use Docker containerization |
| 3 | Deployment Complexity | Manual deployment is time-consuming | Use Kubernetes for automated deployment |
| 4 | Model Monitoring | Hard to track model performance in production | Use Prometheus and Grafana |
| 5 | Model Drift | Model accuracy decreases over time | Implement automated retraining |
| 6 | Scalability | System cannot handle high traffic | Use Kubernetes auto-scaling |
| 7 | CI/CD Integration | Manual testing and deployment | Use GitHub Actions CI/CD |
| 8 | Data Security | Healthcare data privacy issues | Use secure storage and access control |

## 6. Future Research Direction

Although the proposed MLOps framework for colorectal cancer prediction demonstrates high accuracy and efficient deployment, there are several areas for future research and improvement. The current study focuses on histopathological image data and a single machine learning pipeline; however, future work can extend this framework to more advanced and real-world healthcare applications.

One important future research direction is the integration of multi-modal data such as clinical data, genomic data, and radiology images along with histopathological images. Combining multiple data sources can improve prediction accuracy and provide better clinical decision support systems.

Another future direction is the implementation of federated learning in the MLOps pipeline. In healthcare, data privacy is a major concern, and hospitals may not be able to share patient data directly. Federated learning allows machine learning models to be trained across multiple hospitals without sharing raw data, ensuring data privacy and security.

Future research can also focus on edge deployment of colorectal cancer prediction models using lightweight Kubernetes distributions such as K3s. This would allow real-time cancer prediction in remote or resource-limited healthcare environments where cloud infrastructure is not available.

Another important research direction is model explainability and interpretability. In healthcare applications, doctors need to understand why a model makes a particular prediction. Future work can include advanced explainable AI techniques such as SHAP (SHapley Additive Explanations), LIME (Local Interpretable Model-Agnostic Explanations), and uncertainty quantification methods to improve trust and transparency in machine learning models.

Finally, future research can focus on automated model retraining and self-healing MLOps systems, where the system automatically detects data drift, retrains the model, validates performance, and redeploys the updated model without human intervention. This will make the healthcare prediction system fully automated and production-ready.

## 7. Conclusion

This research presented an end-to-end MLOps framework for colorectal cancer prediction using DagsHub, Docker, and Kubernetes (Minikube). The main objective of this study was to bridge the gap between machine learning model development and real-world deployment by implementing an automated, scalable, and reproducible machine learning pipeline for healthcare applications. The proposed system integrates data preprocessing, model training, model evaluation, containerization, deployment, continuous integration/continuous deployment (CI/CD), and monitoring into a single automated workflow.

The machine learning model developed for colorectal cancer prediction achieved high performance in terms of accuracy, precision, recall, F1-score, and AUC, indicating that the model is effective in identifying cancerous and non-cancerous cases. In addition to model performance, the implementation of MLOps tools significantly improved deployment efficiency, system scalability, environment consistency, and monitoring capability compared to traditional machine learning deployment methods.

DagsHub was used for data version control and experiment tracking, which improved reproducibility and collaboration. Docker containerization ensured that the model could run consistently across different environments without dependency issues. Kubernetes (Minikube) enabled automated deployment, load balancing, and auto-scaling, which improved system performance and reliability. The CI/CD pipeline automated testing and deployment processes, reducing deployment time and manual errors. Monitoring tools helped track model performance and system performance, enabling continuous improvement through automated retraining.

The results of this research demonstrate that integrating MLOps with machine learning provides a practical and scalable solution for deploying healthcare prediction models in real-world environments. The proposed framework not only improves model performance but also ensures that the system is reliable, scalable, and production-ready. This research contributes to the field of healthcare artificial intelligence by providing a structured approach for deploying machine learning models using MLOps practices.

**References**

Alla, S., & Adari, S. K. (2021). What is MLOps? In Beginning MLOps with MLFlow (pp. 79–124). Springer. https://doi.org/10.1007/978-1-4842-6543-2_3

Ashmore, R., Calinescu, R., & Paterson, C. (2021). Assuring the machine learning lifecycle: Desiderata, methods, and challenges. ACM Computing Surveys, 54(5), 1–39. https://doi.org/10.1145/3453444

Boettiger, C. (2015). An introduction to Docker for reproducible research. ACM SIGOPS Operating Systems Review, 49(1), 71–79. https://doi.org/10.1145/2723872.2723882

Breck, E., Cai, S., Nielsen, E., Salib, M., & Sculley, D. (2017). The ML test score: A rubric for ML production readiness and technical debt reduction. In IEEE International Conference on Big Data (pp. 1123–1132). IEEE. https://doi.org/10.1109/BigData.2017.8258038

Clissa, L., Macaluso, A., Morelli, R., Occhinegro, A., Piscitiello, E., Taddei, L., Luppi, M., Amici, R., Cerri, M., Hitrec, T., Rinaldi, L., & Zoccoli, A. (2024). Fluorescent neuronal cells v2: Multi-task, multi-format annotations for deep learning in microscopy. Scientific Data, 11, Article 185. https://doi.org/10.1038/s41597-024-03027-7

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M., & Thrun, S. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, 542(7639), 115–118.

Fujii, T. Y., Hayashi, V. T., Arakaki, R., Ruggiero, W. V., Bulla, R., Jr., Hayashi, F. H., & Khalil, K. A. (2021). A digital twin architecture model applied with MLOps techniques to improve short-term energy consumption prediction. Machines, 10(1), Article 23. https://doi.org/10.3390/machines10010023

Garg, S., Pundir, P., Rathee, G., Gupta, P. K., Garg, S., & Ahlawat, S. (2021). On continuous integration/continuous delivery for automated deployment of machine learning models using MLOps. In IEEE International Conference on Artificial Intelligence and Knowledge Engineering (AIKE) (pp. 25–28). https://doi.org/10.1109/AIKE52691.2021.00011

Granlund, T., Stirbu, V., & Mikkonen, T. (2021). Towards regulatory-compliant MLOps. SN Computer Science, 2(5), 1–14. https://doi.org/10.1007/s42979-021-00794-7

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

Kather, J. N., Krisam, J., Charoentong, P., et al. (2019). Predicting survival from colorectal cancer histology slides using deep learning. PLoS Medicine, 16(1).

ISSN: 2583-7354

Khattak, F. K., Subasri, V., Krishnan, A., Dolatabadi, E., Pandya, D., Seyyed-Kalantari, L., & Rudzicz, F. (2023). MLHOps: Machine learning for healthcare operations. arXiv preprint. https://arxiv.org/abs/2305.02474

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A., Ciompi, F., Ghafoorian, M., & Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. Medical Image Analysis, 42, 60–88.

Merkel, D. (2014). Docker: Lightweight Linux containers for consistent development and deployment. Linux Journal, 2014(239).

Moreschini, S., Lomio, F., Hästbacka, D., & Taibi, D. (2022). MLOps for AI-intensive systems. In IEEE SANER Conference (pp. 1293–1294).

Raj, E. (2021). Engineering MLOps. Packt Publishing.

Recupito, G., Pecorelli, F., Catolino, G., Moreschini, S., Di Nucci, D., Palomba, F., & Tamburri, D. A. (2022). A literature review of MLOps tools. In IEEE SEAA Conference (pp. 84–91).

Sculley, D., Holt, G., Golovin, D., Davydov, E., Phillips, T., Ebner, D., Chaudhary, V., Young, M., Crespo, J.-F., & Dennison, D. (2015). Hidden technical debt in ML systems. NeurIPS.

Symeonidis, G., Nerantzis, E., Kazakis, A., & Papakostas, G. A. (2022). MLOps—Definitions, tools and challenges. arXiv preprint.

Testi, M., Ballabio, M., Frontoni, E., Iannello, G., Moccia, S., Soda, P., & Vessio, G. (2022). MLOps: A taxonomy and methodology. IEEE Access, 10, 63606–63618.

Treveil, M. (2020). Introducing MLOps. O'Reilly Media.

Zaharia, M., Chen, A., Davidson, A., Ghodsi, A., Hong, S. A., Konwinski, A., Murching, S., Nykodym, T., Ogilvie, P., Parkhe, M., & Xie, F. (2018). Accelerating the ML lifecycle with MLflow. IEEE Data Engineering Bulletin, 41(4), 39–45.

**Cite this article as:** M.D.R. Siva Santosh et al (2025). Colorectal Cancer Prediction Using MLOps Implemented with DagsHub, Docker, and Kubernetes (Minikube). International Journal of Emerging Knowledge Studies. 4(9), pp.1554– 1569. https://doi.org/10.70333/ijeks-04-12-013