Serverless Machine Learning Model Deployment with AWS SageMaker

*A Project Based Learning Report Submitted in partial fulfilment of the requirements for the award of the degree*

*of*

**Bachelor of Technology**

**in the Department of Computer Science & Engineering**

**Cloud Based AI/ML Speciality (22SDCS07A)**

Submitted by

**2210030223: GUDIPUDI LOHITH KUMAR**

Under the guidance of

**Ms. P. Sree Lakshmi**



Department of Computer Science and Engineering

Koneru Lakshmaiah Education Foundation, Aziz Nagar

Aziz Nagar – 500075

March - 2025.

**Introduction**

This project focuses on the serverless deployment of machine learning (ML) models using AWS SageMaker, a powerful cloud-based ML platform. With the rise of cloud computing, organizations increasingly seek scalable and cost-efficient solutions for ML deployment. Traditional methods involve complex infrastructure management, making deployment a challenge for teams with limited resources. This project eliminates such complexity by leveraging serverless architecture. It allows developers to focus on building and optimizing ML models without worrying about infrastructure provisioning or scaling. The implementation highlights how serverless ML can streamline workflows, reduce operational costs, and support real-time model predictions in a reliable and scalable environment.

The core AWS services used in this project are Amazon SageMaker, AWS Lambda, API Gateway, and Amazon S3. SageMaker enables model training, tuning, and deployment without managing servers. Lambda is used to trigger real-time inferences and connect with SageMaker Endpoints. API Gateway acts as a secure interface to expose the model as an API, while S3 handles data storage. These services together ensure a fully serverless ML pipeline. The selection of these tools is justified by their ability to scale automatically, reduce infrastructure overhead, and integrate seamlessly, making them ideal for cost-effective, production-ready machine learning deployment.

The purpose of this project is to demonstrate an end-to-end serverless machine learning deployment using AWS services. The expected outcome includes a scalable, cost-efficient ML model accessible via API for real-time predictions. This setup aims to reduce operational complexity while showcasing best practices in modern cloud-based ML deployment.

**Methodology**

The architecture for serverless machine learning deployment using AWS SageMaker involves several interconnected components. First, raw data is stored and accessed from Amazon S3, which serves as the central data repository. The data is preprocessed either locally or using SageMaker’s built-in processing capabilities. The cleaned data is then used to train an ML model using SageMaker Training Jobs. Once the model is trained and validated, it is deployed to a SageMaker Endpoint, enabling real-time inference.

For inference, a client or web application sends a request to an API Gateway, which routes the request to an AWS Lambda function. The Lambda function, in turn, invokes the SageMaker Endpoint to get predictions. The response is returned to the client via the API Gateway. This serverless workflow ensures scalability, reduced latency, and easy integration, making it ideal for real-time applications. Monitoring and logging are handled through Amazon CloudWatch for debugging and performance tracking.

In this architecture, each AWS service plays a specific role and communicates seamlessly with others. Amazon S3 stores the input datasets and trained model artifacts. AWS SageMaker accesses this data for model training and writes back the model output to S3. Once the model is deployed, SageMaker Endpoints remain active to serve inference requests.

API Gateway acts as a public-facing REST API that receives requests from users or applications. These requests are passed to AWS Lambda, which contains lightweight logic to parse input, invoke the SageMaker Endpoint, and return the output. Lambda communicates with SageMaker using the boto3 SDK and maintains stateless interactions, allowing for rapid scaling.

CloudWatch captures logs from Lambda and SageMaker for observability. This interaction between services ensures a modular, maintainable system that adheres to serverless principles while offering a high-performance ML inference pipeline.

AWS services were selected for their scalability, seamless integration, and serverless capabilities. SageMaker handles model lifecycle, Lambda supports real-time compute without provisioning, and API Gateway securely exposes endpoints. Combined with S3 for storage and CloudWatch for monitoring, these services provide an efficient and fully managed ML deployment environment.

**Implementation Steps**

The infrastructure setup for serverless machine learning deployment begins with configuring Amazon S3 to store training data, model artifacts, and logs. After creating the required S3 buckets, Amazon SageMaker is configured to use this data for model training. A SageMaker notebook instance or Studio is launched to develop and train the ML model. Once the model is trained, a SageMaker Endpoint is created to serve real-time inference requests.

Next, API Gateway is set up to expose the model as a REST API, followed by the creation of an AWS Lambda function that invokes the SageMaker endpoint when triggered by API calls. The Lambda function is configured with minimal logic to handle input and output transformations.

Finally, Amazon CloudWatch is configured to monitor logs and metrics for Lambda, SageMaker, and API Gateway, providing full visibility into the pipeline’s performance. This infrastructure is scalable, fault-tolerant, and requires no server management.

Security is a critical aspect of this deployment. AWS Identity and Access Management (IAM) roles are created to grant least-privilege permissions to services. For instance, the SageMaker execution role has permissions to read/write from Amazon S3 and invoke logging actions in CloudWatch. Similarly, the Lambda function has permissions to invoke SageMaker endpoints and access specific environment variables.

Custom IAM policies are attached to control data access strictly, ensuring only authorized services and users can access sensitive information. Role-based access controls (RBAC) are enforced to separate responsibilities, such as development, deployment, and monitoring.

API Gateway is configured with authorization tokens or API keys to prevent unauthorized access. Optional VPC endpoints can be used to ensure private connectivity between services.

CloudTrail is used to log all activities across services, and encryption at rest (via AWS KMS) and in transit (HTTPS) ensures data privacy and integrity throughout the workflow.

To streamline model updates and deployments, a basic CI/CD pipeline can be integrated using AWS CodePipeline and CodeBuild. Whenever a new model version or Lambda function code is pushed to a version control system like GitHub, the pipeline triggers automated testing, validation, and deployment.

For SageMaker, model package groups can track versioned models and automate updates to endpoints. Using infrastructure-as-code tools like AWS CloudFormation or Terraform, the entire architecture can be deployed and managed efficiently, reducing manual intervention. This automation ensures consistency, faster rollbacks, and continuous delivery of improved ML models in production.