



MLOps를 위한 FSx ONTAP

NetApp artificial intelligence solutions

NetApp
February 12, 2026

목차

MLOps를 위한 FSx ONTAP	1
MLOps용 Amazon FSx for NetApp ONTAP (FSx ONTAP)	1
1부 - Amazon FSx for NetApp ONTAP (FSx ONTAP)을 AWS SageMaker에 개인 S3 버킷으로 통합	1
소개	1
사용자 가이드	1
유용한 디버깅 체크리스트	14
FAQ (2023년 9월 27일 기준)	15
2부 - SageMaker에서 모델 학습을 위한 데이터 소스로 AWS Amazon FSx for NetApp ONTAP (FSx ONTAP) 활용	15
소개	15
FSx ONTAP 무엇입니까?	15
필수 조건	15
통합 개요	16
단계별 통합	17
3부 - 단순화된 MLOps 파이프라인(CI/CT/CD) 구축	24
소개	24
명백한	24
필수 조건	25
아키텍처	25
단계별 구성	25

MLOps를 위한 FSx ONTAP

MLOps용 Amazon FSx for NetApp ONTAP (FSx ONTAP)

이 섹션에서는 AI 인프라 개발의 실제 적용을 깊이 있게 살펴보고 FSx ONTAP 사용하여 MLOps 파이프라인을 구성하는 방법에 대한 종단 간 연습을 제공합니다. 세 가지 포괄적인 예로 구성된 이 문서는 강력한 데이터 관리 플랫폼을 통해 MLOps 요구 사항을 충족하는 방법을 안내합니다.

이 기사에서는 다음 주제에 대해 다룹니다.

1. "1부 - Amazon FSx for NetApp ONTAP (FSx ONTAP)을 AWS SageMaker에 개인 S3 버킷으로 통합"
2. "2부 - SageMaker에서 모델 학습을 위한 데이터 소스로 Amazon FSx for NetApp ONTAP (FSx ONTAP) 활용"
3. "3부 - 단순화된 MLOps 파이프라인(CI/CT/CD) 구축"

이 섹션을 마치면 FSx ONTAP 사용하여 MLOps 프로세스를 간소화하는 방법을 확실히 이해하게 될 것입니다.

1부 - Amazon FSx for NetApp ONTAP (FSx ONTAP)을 AWS SageMaker에 개인 S3 버킷으로 통합

이 섹션에서는 AWS SageMaker를 사용하여 FSx ONTAP 프라이빗 S3 버킷으로 구성하는 방법에 대한 가이드를 제공합니다.

소개

이 페이지에서는 SageMaker를 예로 들어 FSx ONTAP 개인 S3 버킷으로 구성하는 방법에 대한 지침을 제공합니다.

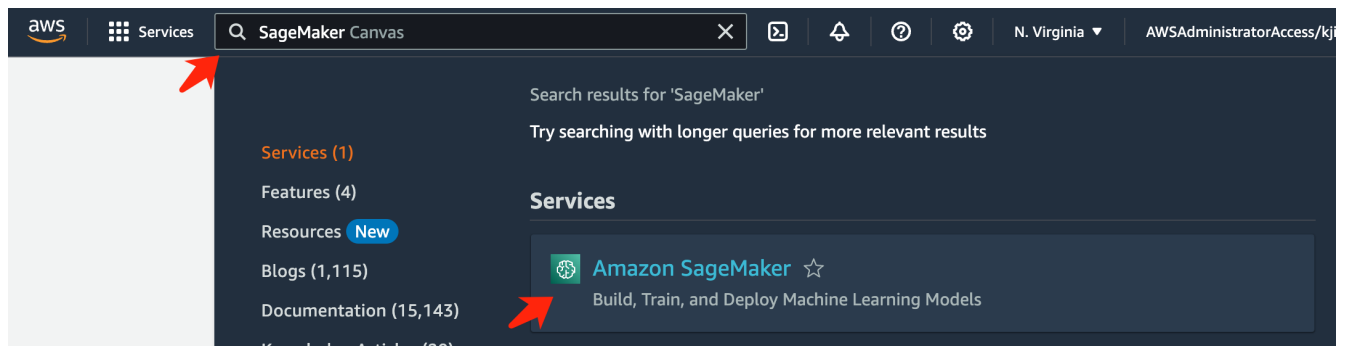
FSx ONTAP 에 대한 자세한 내용은 이 프레젠테이션을 참조하세요.["비디오 링크"](#))

사용자 가이드

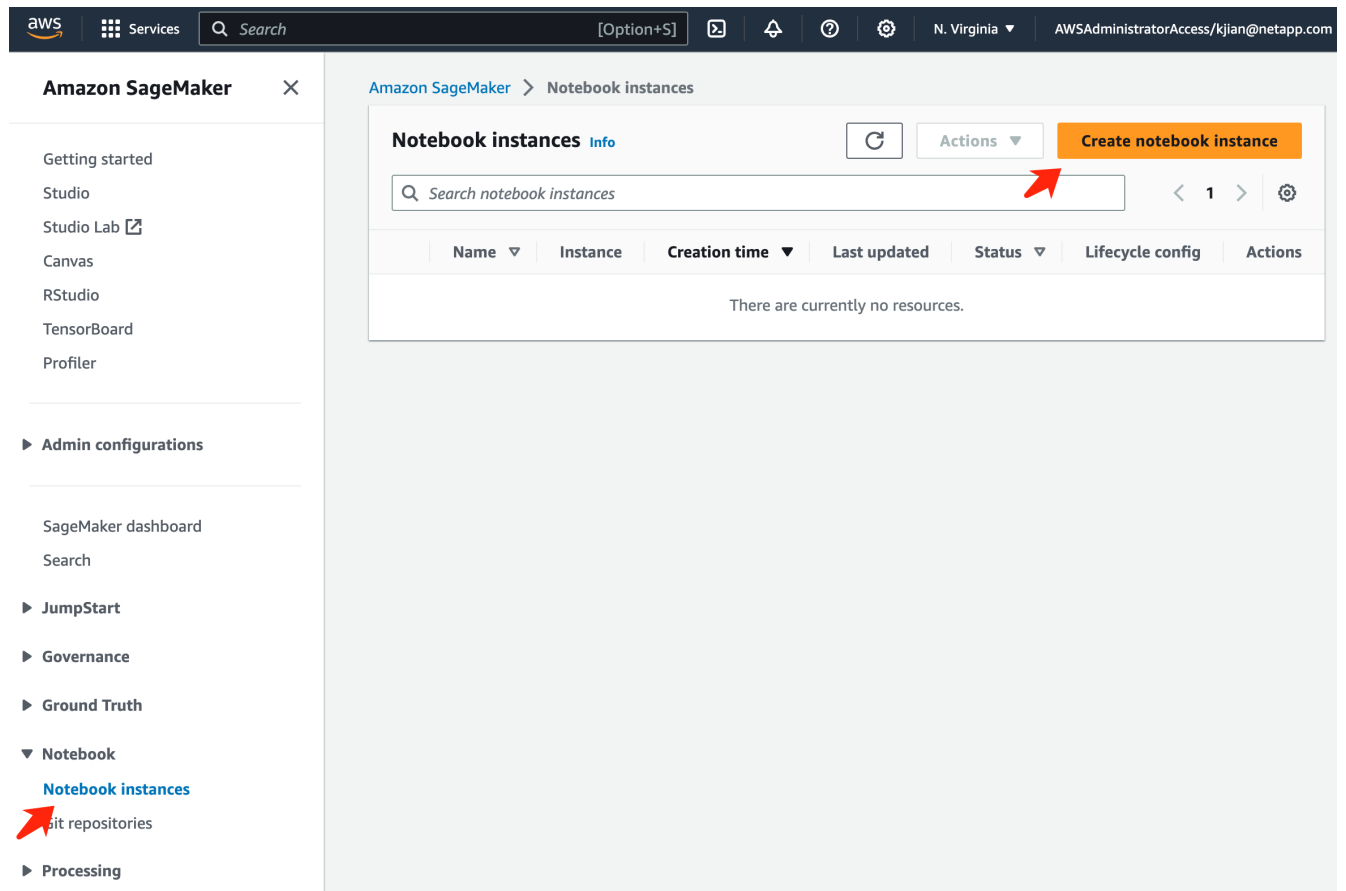
서버 생성

SageMaker Notebook 인스턴스 생성

1. AWS 콘솔을 엽니다. 검색 패널에서 SageMaker를 검색하고 **Amazon SageMaker** 서비스를 클릭합니다.



2. 노트북 탭에서 *노트북 인스턴스*를 열고 주황색 버튼인 *노트북 인스턴스 만들기*를 클릭합니다.



3. 생성 페이지에서 노트북 인스턴스 이름*을 입력합니다. *네트워크 패널을 확장합니다. 다른 항목은 기본값으로 두고 **VPC**, 서브넷, *보안 그룹*을 선택합니다. (이 *VPC*와 *서브넷*은 나중에 FSx ONTAP 파일 시스템을 만드는 데 사용됩니다.) 오른쪽 하단에 있는 주황색 버튼 *노트북 인스턴스 만들기*를 클릭합니다.

Amazon SageMaker > Notebook instances > Create notebook instance

Create notebook instance

Amazon SageMaker provides pre-built fully managed notebook instances that run Jupyter notebooks. The notebook instances include example code for common model training and hosting exercises. [Learn more](#)

Notebook instance settings

Notebook instance name
fsxn-demo
Maximum of 63 alphanumeric characters. Can include hyphens (-), but not spaces. Must be unique within your account in an AWS Region.

Notebook instance type
ml.t3.medium

Elastic Inference [Learn more](#)
none

Platform identifier [Learn more](#)
Amazon Linux 2, Jupyter Lab 3

▶ Additional configuration

Permissions and encryption

IAM role
Notebook instances require permissions to call other services including SageMaker and S3. Choose a role or let us create a role with the [AmazonSageMakerFullAccess](#) IAM policy attached.
AmazonSageMakerServiceCatalogProductsUseRole

Create role using the role creation wizard

Root access - optional
☒ Enable - Give users root access to the notebook.
☐ Disable - Don't give users root access to the notebook.
Lifecycle configurations always have root access.

Encryption key - optional
 Encrypt your notebook data. Choose an existing KMS key or enter a key's ARN.
 No Custom Encryption

▼ Network - optional

VPC - optional
Default vpc-0df3956ab1fca2ec9 (172.31.0.0/16)

Subnet
Choose a subnet in an availability zone supported by Amazon SageMaker.
 subnet-00060df0d9f562672 (172.31.16.0/20) | us-east-1a

Security group(s)
 sg-0a39b3985770e9256 (default) X

Direct internet access
☒ Enable — Access the internet directly through Amazon SageMaker.
☐ Disable — Access the internet through a VPC.
To train or host models from a notebook, you need internet access. To enable internet access, make sure that your VPC has a NAT gateway and your security group allows outbound connections. [Learn more](#)

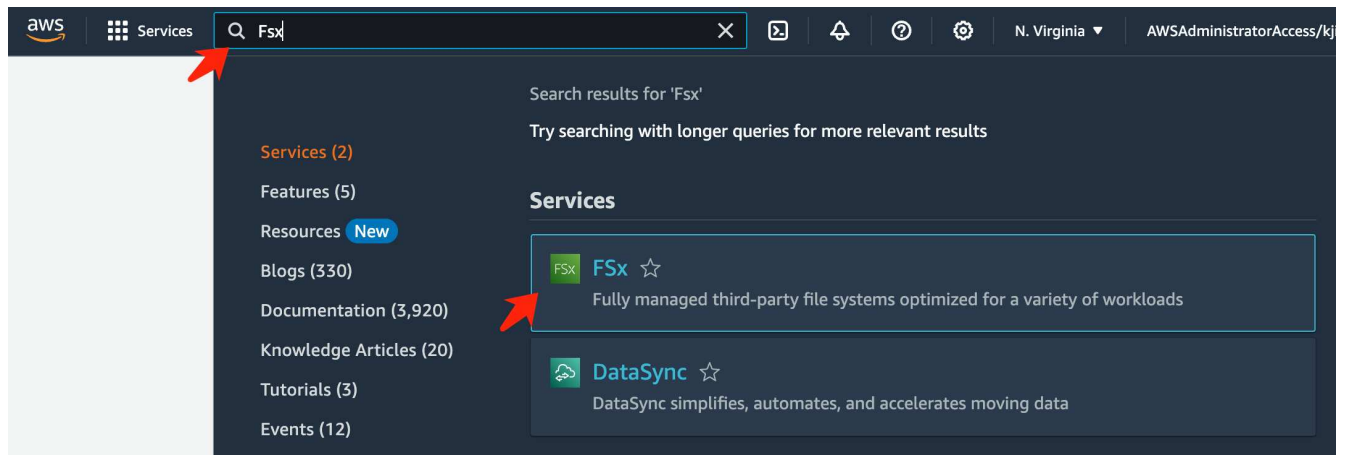
▶ Git repositories - optional

▶ Tags - optional

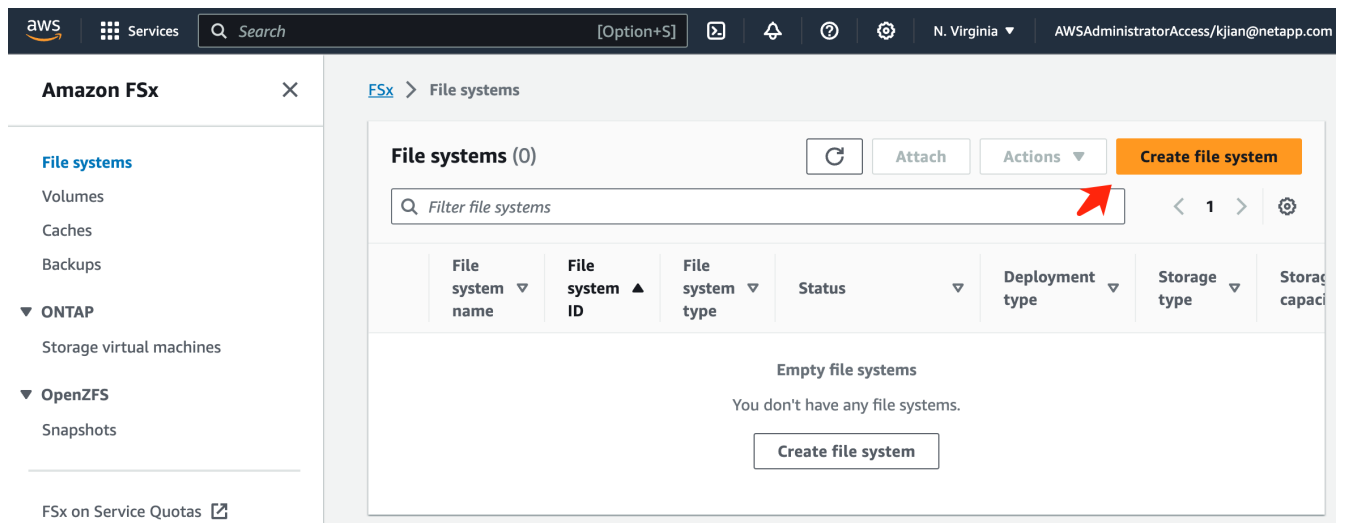
Cancel Create notebook instance

FSx ONTAP 파일 시스템 생성

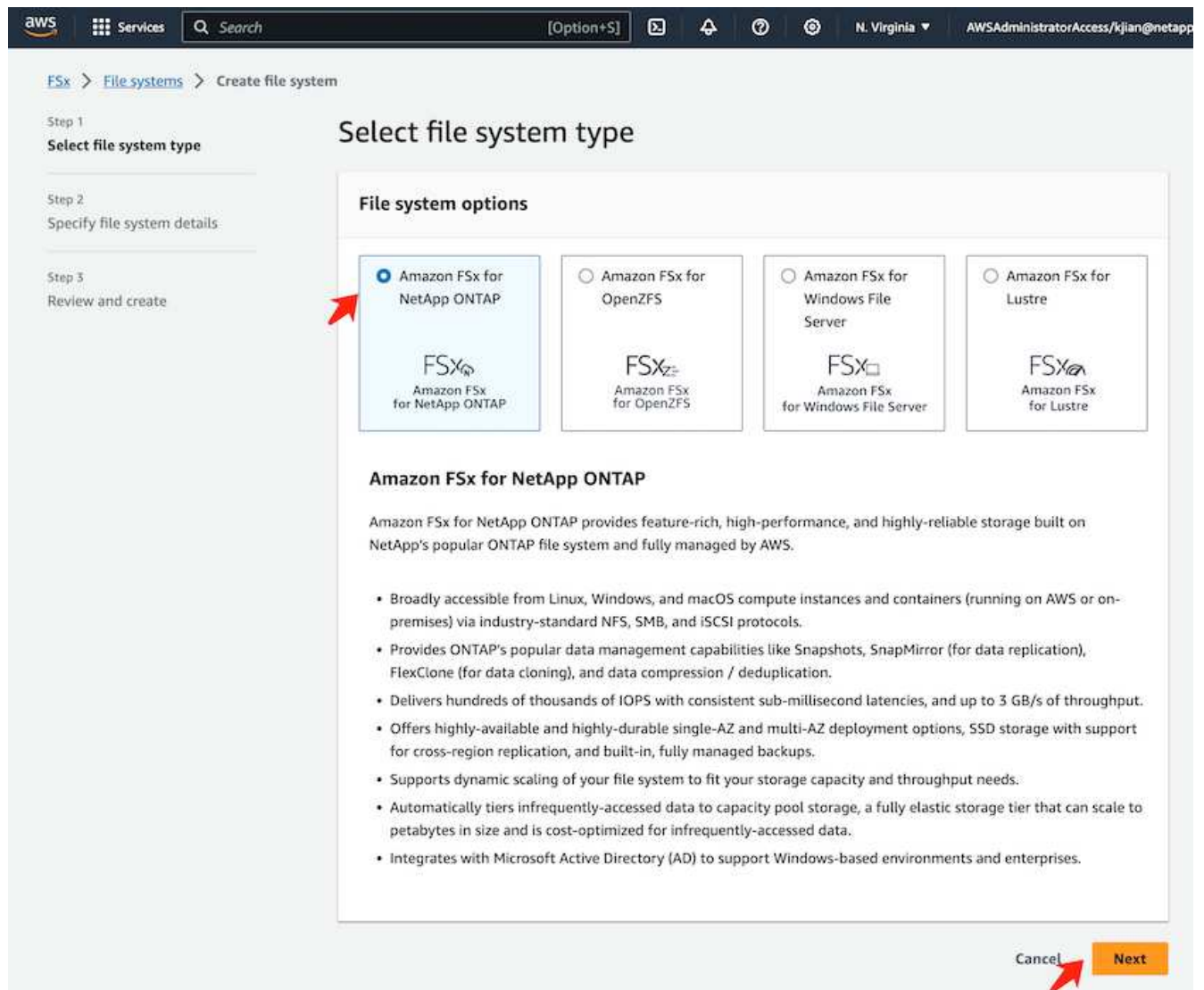
1. AWS 콘솔을 엽니다. 검색 패널에서 Fsx를 검색하고 서비스 *FSx*를 클릭합니다.



2. *파일 시스템 만들기*를 클릭합니다.

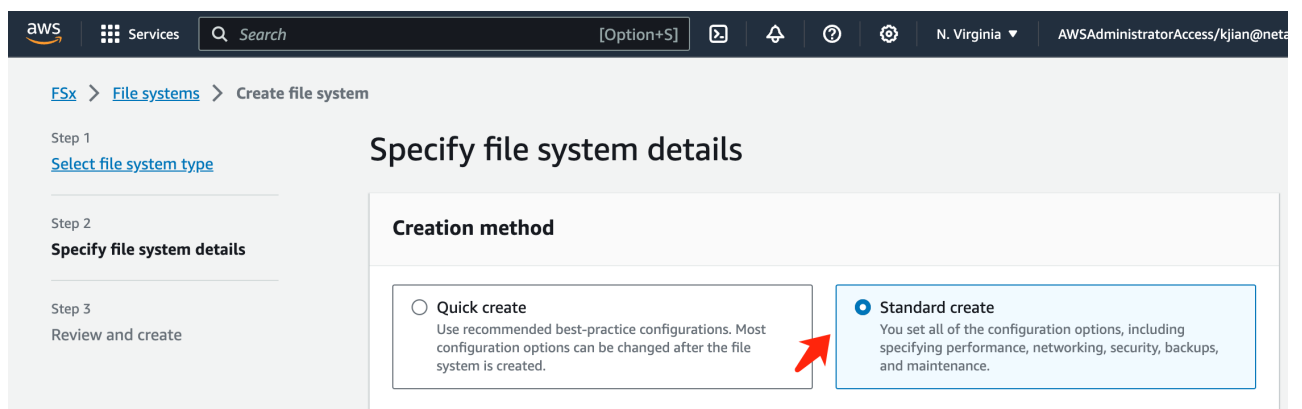


3. 첫 번째 카드인 *FSx ONTAP*을 선택하고 *다음*을 클릭합니다.



4. 세부 정보 구성 페이지에서.

a. 표준 생성 옵션을 선택하세요.



b. *파일 시스템 이름*과 *SSD 저장 용량*을 입력하세요.

File system details

File system name - optional

Info

fsxn-demo

Maximum of 256 Unicode letters, whitespace, and numbers, plus + - = . _ : /

Deployment type

Info

☒ Multi-AZ

☐ Single-AZ

SSD storage capacity

Info

1024

GiB

Minimum 1024 GiB; Maximum 192 TiB.

Provisioned SSD IOPS

Amazon FSx provides 3 IOPS per GiB of storage capacity. You can also provision additional SSD IOPS as needed.

☒ Automatic (3 IOPS per GiB of SSD storage)

☐ User-provisioned

Throughput capacity

Info

The sustained speed at which the file server hosting your file system can serve data. The file server can also burst to higher speeds for periods of time.

☒ Recommended throughput capacity

128 MB/s

☐ Specify throughput capacity

c. **SageMaker Notebook** 인스턴스에 *VPC*와 *서브넷*을 동일하게 사용해야 합니다.

Network & security

Virtual Private Cloud (VPC) [Info](#)

Specify the VPC from which your file system is accessible.

vpc-0df3956ab1fca2ec9 (CIDR: 172.31.0.0/16) ▼

VPC Security Groups [Info](#)

Specify VPC Security Groups to associate with your file system's network interfaces.

Choose VPC security group(s) ▼

sg-0a39b3985770e9256 (default) ✕

Preferred subnet [Info](#)

Specify the preferred subnet for your file system.

subnet-00060df0d0f562672 (us-east-1a | use1-az4) ▼

Standby subnet

subnet-02b029f24d03a4af2 (us-east-1b | use1-az6) ▼

VPC route tables [Info](#)

Specify the VPC route tables to associate with your file system.

- ☒ VPC's main route table
- ☐ Select one or more VPC route tables

Endpoint IP address range [Info](#)

Specify the IP address range in which the endpoints to access your file system will be created

- ☒ Unallocated IP address range from your VPC
Simplest option for access from other AWS services or peered / on-premises networks
- ☐ Floating IP address range outside your VPC
- ☐ Enter an IP address range

d. SVM(스토리지 가상 머신)에 대한 스토리지 가상 머신 이름을 입력하고 *암호를 지정*합니다.

Default storage virtual machine configuration

Storage virtual machine name [Info](#)

fsxn-svm-demo

SVM administrative password
Password for this SVM's "vsadmin" user, which you can use to access the ONTAP CLI or REST API. You can provide a password later if you don't provide one now.

☐ Don't specify a password

☒ Specify a password

Password

.....

Confirm password

.....

Volume security style
The security style of the volume determines whether preference is given to NTFS or UNIX ACLs for multi-protocol access. The MIXED mode is not required for multi-protocol access and is only recommended for advanced users.

Unix (Linux) ▼

Active Directory
Joining an Active Directory enables access from Windows and MacOS clients over the SMB protocol.

☒ Do not join an Active Directory

☐ Join an Active Directory

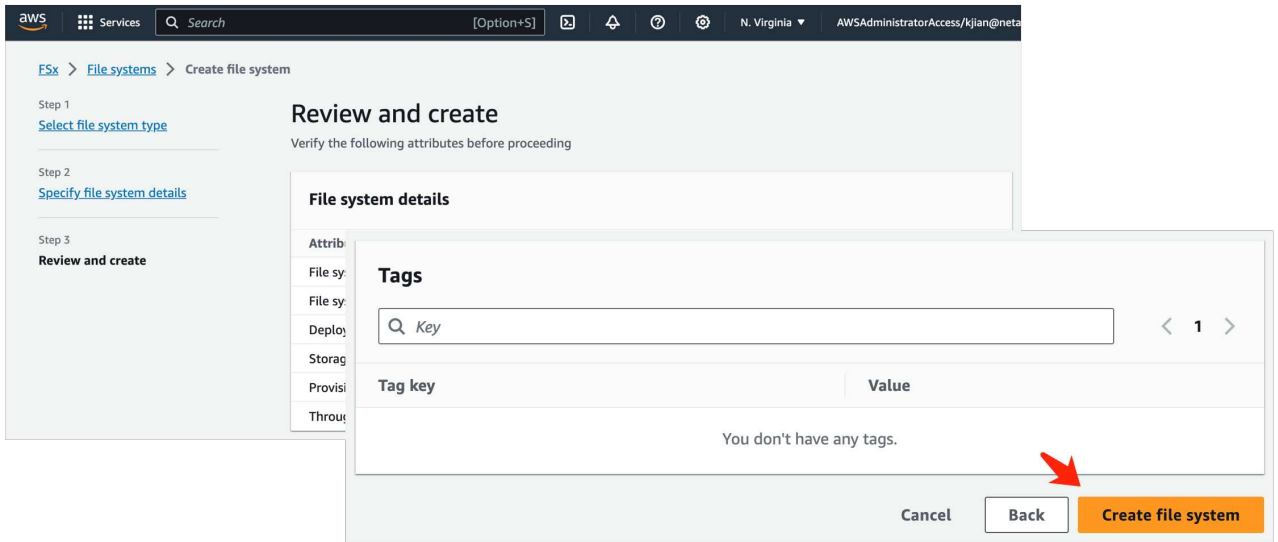
e. 다른 항목은 기본값으로 두고 오른쪽 하단에 있는 주황색 버튼 *다음*을 클릭하세요.

► **Backup and maintenance - optional**

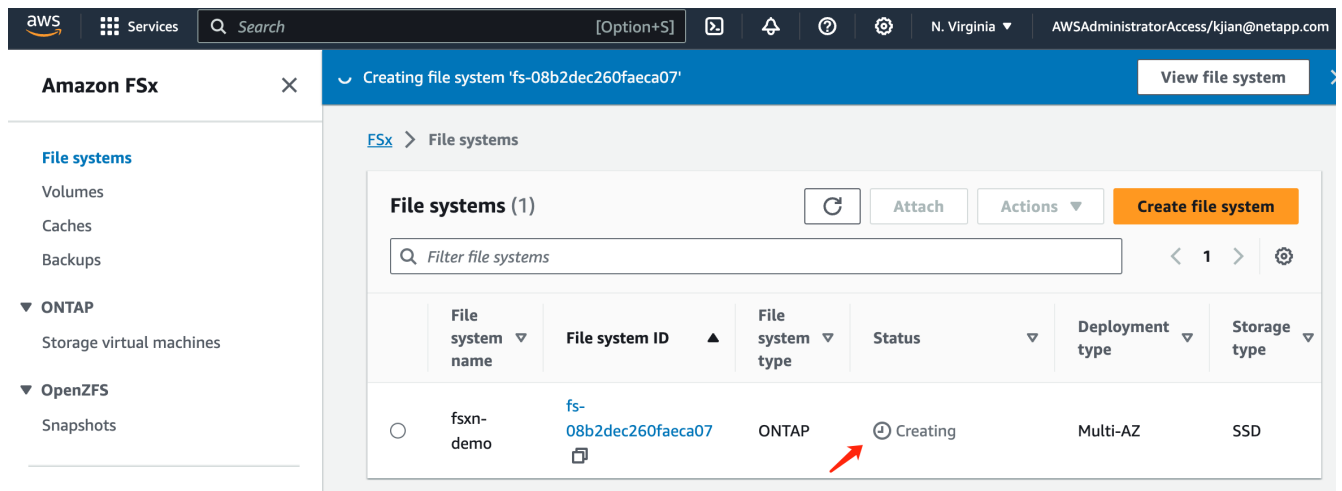
► **Tags - optional**

Cancel **Back** **Next**

f. 검토 페이지 오른쪽 하단에 있는 주황색 버튼 *파일 시스템 만들기*를 클릭하세요.



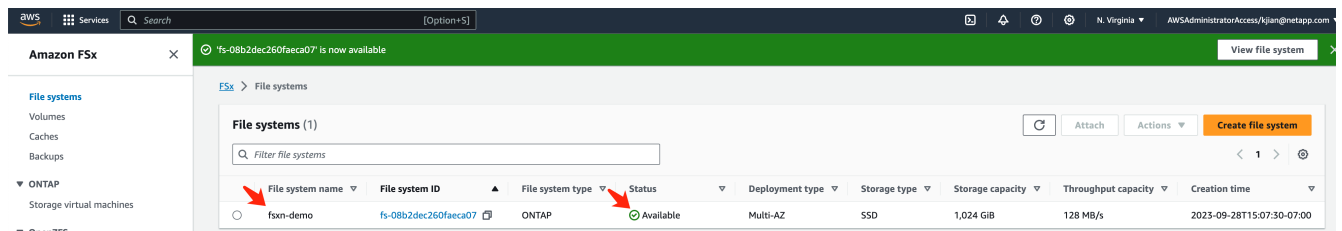
5. FSx 파일 시스템을 시작하는 데 약 *20~40분*이 걸릴 수 있습니다.



서버 구성

ONTAP 구성

1. 생성된 FSx 파일 시스템을 엽니다. 상태가 *사용 가능*인지 확인하세요.



2. 관리 탭을 선택하고 *관리 엔드포인트 - IP 주소*와 * ONTAP 관리자 사용자 이름*을 그대로 둡니다.

Amazon FSx fsxn-demo (fs-08b2dec260faeca07)

Summary

File system ID fs-08b2dec260faeca07	SSD storage capacity 1024 GiB	Availability Zones us-east-1a (Preferred) us-east-1b (Standby)
Lifecycle state Creating	Throughput capacity 128 MB/s	Creation time 2023-09-28T14:41:50-07:00
File system type ONTAP	Provisioned IOPS 3072	
Deployment type Multi-AZ		

ONTAP administration

Management endpoint - DNS name management.fs-08b2dec260faeca07.fsx.us-east-1.amazonaws.com	Management endpoint - IP address 172.31.255.250	ONTAP administrator username fsxadmin
Inter-cluster endpoint - DNS name intercluster.fs-08b2dec260faeca07.fsx.us-east-1.amazonaws.com	Inter-cluster endpoint - IP address 172.31.31.157	ONTAP administrator password Update

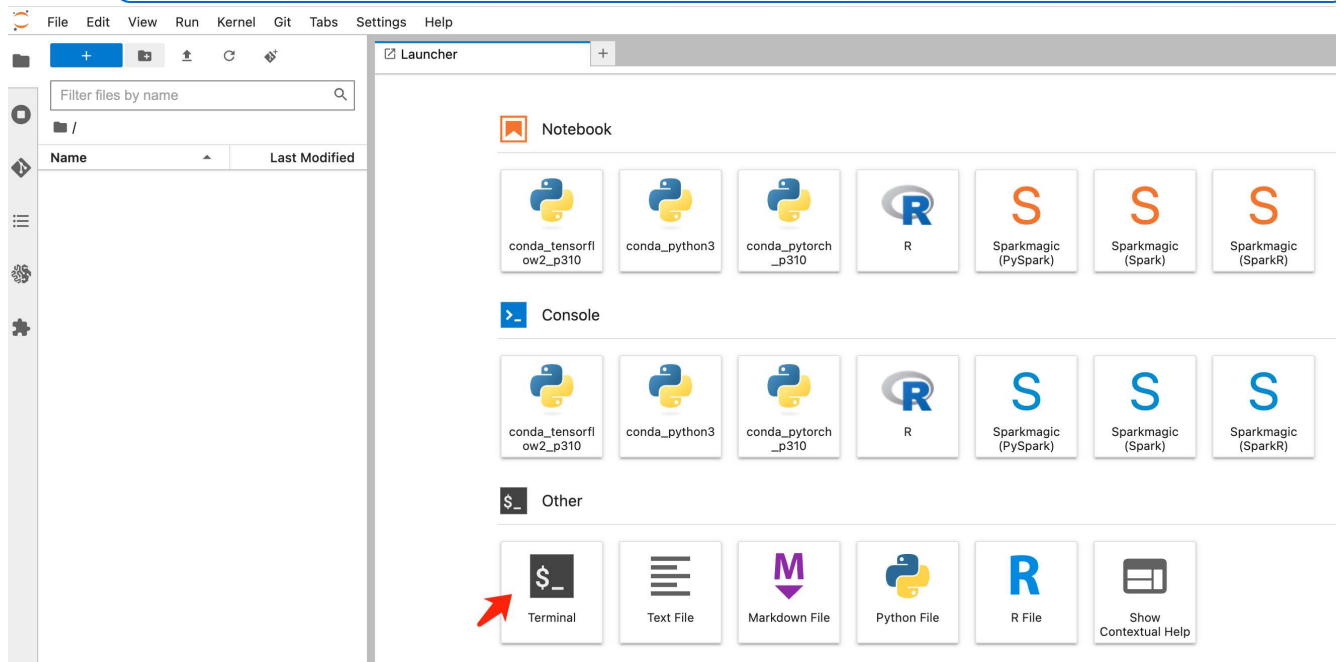
3. 생성된 *SageMaker Notebook 인스턴스*를 열고 *JupyterLab 열기*를 클릭합니다.

Amazon SageMaker Notebook instances

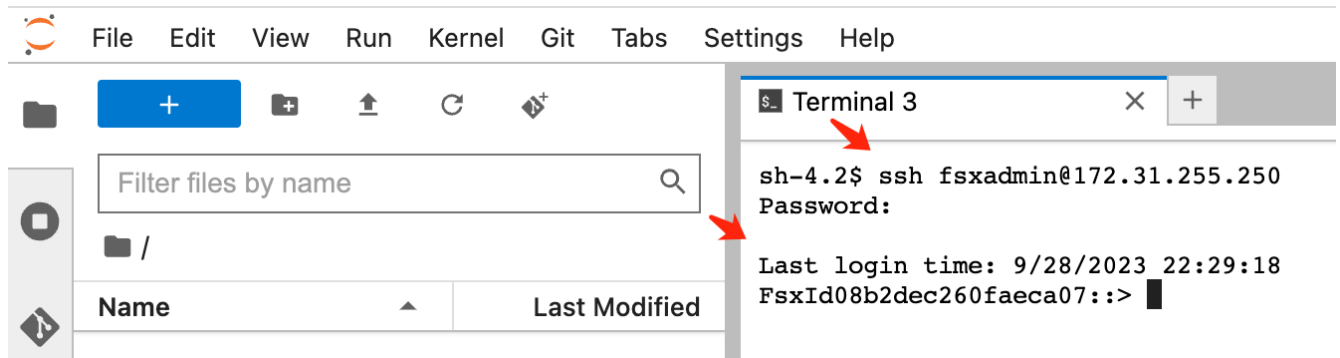
Notebook instances

Name	Instance	Creation time	Last updated	Status	Lifecycle config	Actions
fsxn-demo	ml.t3.medium	9/28/2023, 1:47:27 PM	9/28/2023, 1:50:28 PM	InService		Open Jupyter Open JupyterLab

4. Jupyter Lab 페이지에서 새 *터미널*을 엽니다.



5. FSx ONTAP 파일 시스템에 로그인하려면 ssh 명령어 `ssh <관리자 사용자 이름>@< ONTAP 서버 IP>`를 입력하세요. (사용자 이름과 IP 주소는 2단계에서 검색됩니다.) *스토리지 가상 머신*을 생성할 때 사용한 비밀번호를 사용하세요.



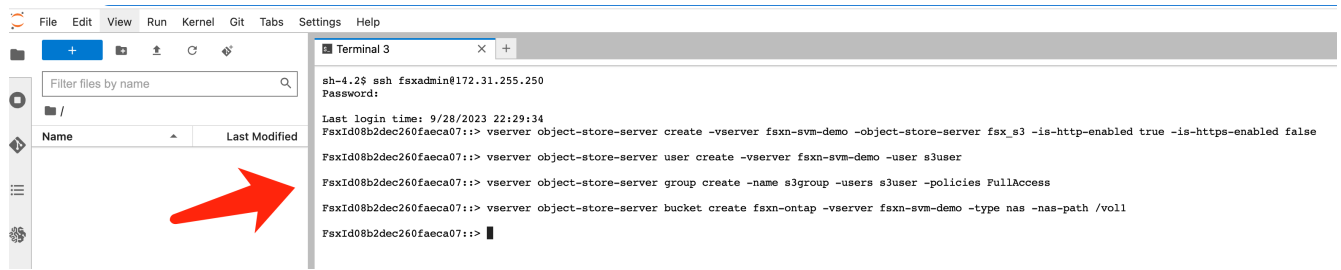
6. 다음 순서대로 명령을 실행하세요. 우리는 *FSx ONTAP 개인 S3 버킷 이름*의 이름으로 *fsxn-ontap*을 사용합니다. -vserver 인수에는 *스토리지 가상 머신 이름*을 사용하세요.

```
vserver object-store-server create -vserver fsxn-svm-demo -object-store
-server fsx_s3 -is-http-enabled true -is-https-enabled false

vserver object-store-server user create -vserver fsxn-svm-demo -user
s3user

vserver object-store-server group create -name s3group -users s3user
-policies FullAccess

vserver object-store-server bucket create fsxn-ontap -vserver fsxn-svm-
demo -type nas -nas-path /vol1
```



7. 아래 명령을 실행하여 FSx ONTAP private S3의 엔드포인트 IP와 자격 증명을 검색합니다.

```

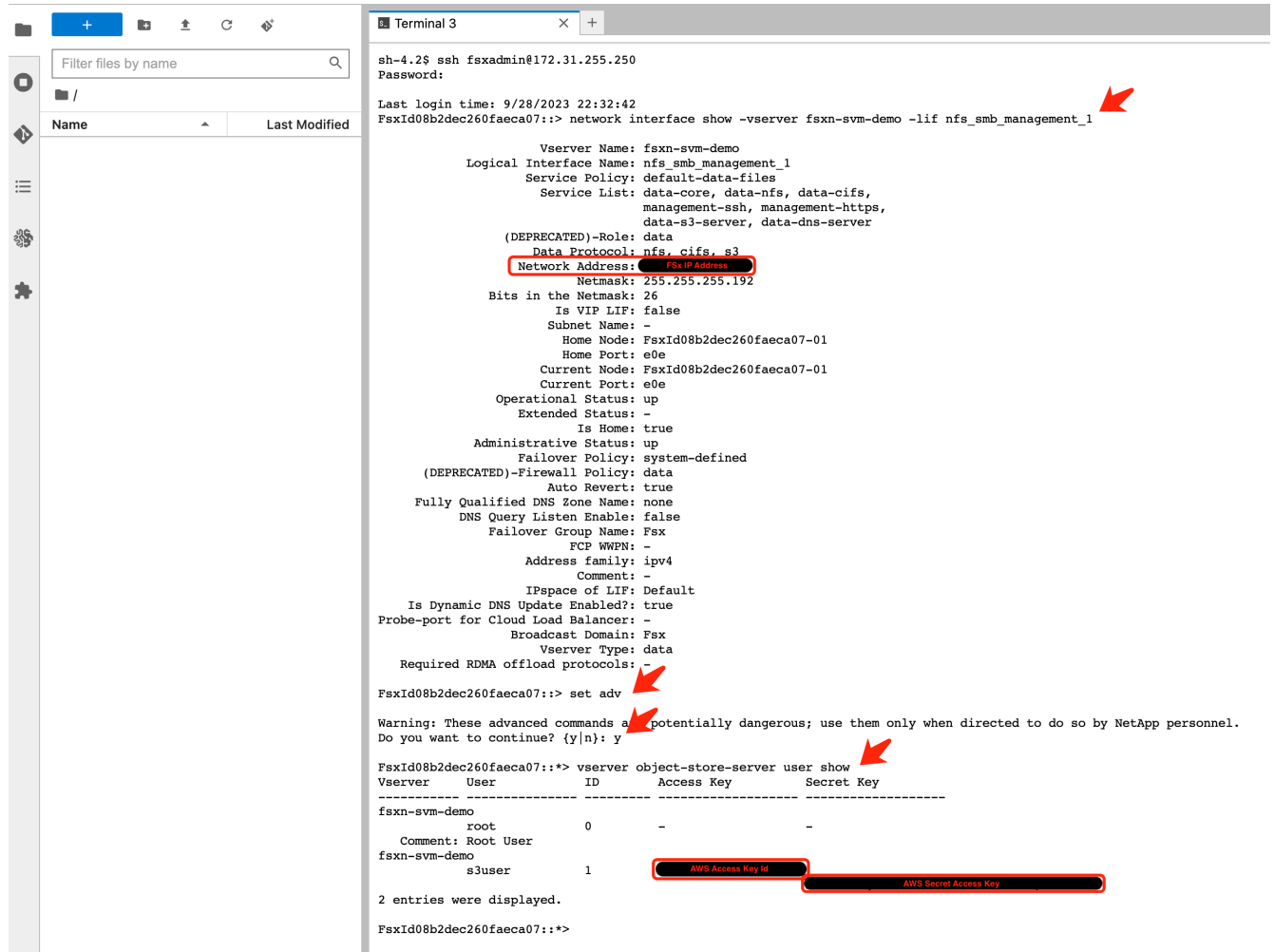
network interface show -vserver fsxn-svm-demo -lif nfs_smb_management_1

set adv

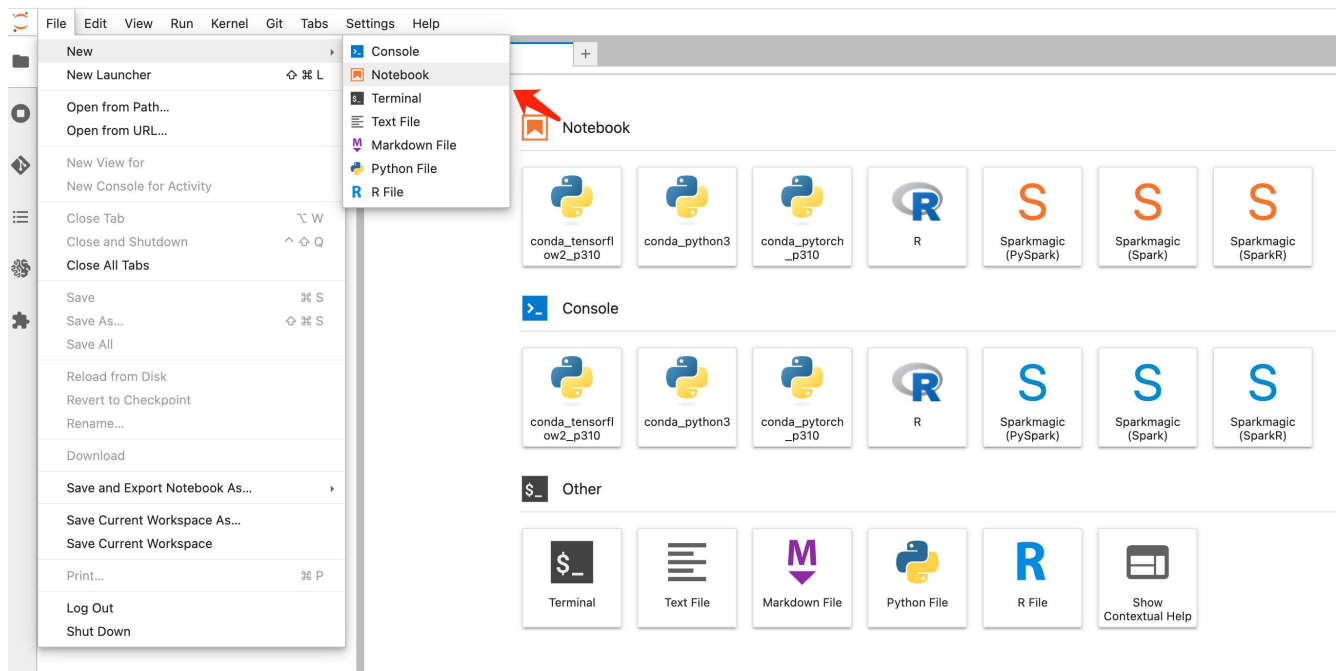
vserver object-store-server user show

```

8. 나중에 사용할 수 있도록 엔드포인트 IP와 자격 증명을 보관하세요.



1. SageMaker Notebook 인스턴스에서 새로운 Jupyter notebook을 만듭니다.

2. 아래 코드를 해결 방법으로 사용하여 FSx ONTAP 개인 S3 버킷에 파일을 업로드하세요. 포괄적인 코드 예제는 이 노트북을 참조하세요. "[fsxn_demo.ipynb](#)"

```
# Setup configurations
# ----- Manual configurations -----
seed: int = 77 # Random
seed
bucket_name: str = 'fsxn-ontap' # The bucket
name in ONTAP
aws_access_key_id = '<Your ONTAP bucket key id>' # Please get
this credential from ONTAP
aws_secret_access_key = '<Your ONTAP bucket access key>' # Please get
this credential from ONTAP
fsx_endpoint_ip: str = '<Your FSx ONTAP IP address>' # Please get
this IP address from FSx ONTAP
# ----- Manual configurations -----

# Workaround
## Permission patch
!mkdir -p vol1
!sudo mount -t nfs $fsx_endpoint_ip:/vol1 /home/ec2-user/SageMaker/vol1
!sudo chmod 777 /home/ec2-user/SageMaker/vol1

## Authentication for FSx ONTAP as a Private S3 Bucket
!aws configure set aws_access_key_id $aws_access_key_id
```

```

!aws configure set aws_secret_access_key $aws_secret_access_key

## Upload file to the FSx ONTAP Private S3 Bucket
%%capture
local_file_path: str = <Your local file path>

!aws s3 cp --endpoint-url http://$fsx_endpoint_ip /home/ec2-user
/SageMaker/$local_file_path s3://$bucket_name/$local_file_path

# Read data from FSx ONTAP Private S3 bucket
## Initialize a s3 resource client
import boto3

# Get session info
region_name = boto3.session.Session().region_name

# Initialize Fsx S3 bucket object
# --- Start integrating SageMaker with FSXN ---
# This is the only code change we need to incorporate SageMaker with
FSXN
s3_client: boto3.client = boto3.resource(
    's3',
    region_name=region_name,
    aws_access_key_id=aws_access_key_id,
    aws_secret_access_key=aws_secret_access_key,
    use_ssl=False,
    endpoint_url=f'http://{fsx_endpoint_ip}',
    config=boto3.session.Config(
        signature_version='s3v4',
        s3={'addressing_style': 'path'}
    )
)
# --- End integrating SageMaker with FSXN ---

## Read file byte content
bucket = s3_client.Bucket(bucket_name)

binary_data = bucket.Object(data.filename).get()['Body']

```

이것으로 FSx ONTAP 과 SageMaker 인스턴스 간의 통합이 완료되었습니다.

유용한 디버깅 체크리스트

- SageMaker Notebook 인스턴스와 FSx ONTAP 파일 시스템이 동일한 VPC에 있는지 확인하세요.
- ONTAP 에서 **set dev** 명령을 실행하여 권한 수준을 *dev*로 설정하는 것을 잊지 마세요.

FAQ (2023년 9월 27일 기준)

질문: FSx ONTAP 에 파일을 업로드할 때 "**CreateMultipartUpload** 작업을 호출하는 동안 오류가 발생했습니다(**NotImplemented**). 요청한 **s3** 명령이 구현되지 않았습니다"라는 오류가 발생하는 이유는 무엇입니까?

답변: FSx ONTAP 개인 S3 버킷으로서 최대 100MB의 파일 업로드를 지원합니다. S3 프로토콜을 사용할 경우, 100MB가 넘는 파일은 100MB 단위로 나누고, 'CreateMultipartUpload' 함수를 호출합니다. 하지만 현재 FSx ONTAP private S3 구현에서는 이 기능을 지원하지 않습니다.

질문: FSx ONTAP 에 파일을 업로드할 때 "**PutObject** 작업을 호출하는 동안 오류가 발생했습니다(**AccessDenied**): 액세스가 거부되었습니다"라는 오류가 발생하는 이유는 무엇입니까?

답변: SageMaker Notebook 인스턴스에서 FSx ONTAP 개인 S3 버킷에 액세스하려면 AWS 자격 증명을 FSx ONTAP 자격 증명으로 전환하세요. 그러나 인스턴스에 쓰기 권한을 부여하려면 버킷을 마운트하고 'chmod' 셸 명령을 실행하여 권한을 변경하는 임시 해결책이 필요합니다.

질문: FSx ONTAP 개인 S3 버킷을 다른 SageMaker ML 서비스와 통합하려면 어떻게 해야 하나요?

A: 안타깝게도 SageMaker 서비스 SDK는 개인 S3 버킷의 엔드포인트를 지정하는 방법을 제공하지 않습니다. 결과적으로 FSx ONTAP S3는 Sagemaker Data Wrangler, Sagemaker Clarify, Sagemaker Glue, Sagemaker Athena, Sagemaker AutoML 등의 SageMaker 서비스와 호환되지 않습니다.

2부 - SageMaker에서 모델 학습을 위한 데이터 소스로 AWS Amazon FSx for NetApp ONTAP (FSx ONTAP) 활용

이 문서는 SageMaker에서 PyTorch 모델을 학습하기 위해 Amazon FSx for NetApp ONTAP (FSx ONTAP)을 사용하는 방법에 대한 튜토리얼로, 특히 타이어 품질 분류 프로젝트에 대한 내용입니다.

소개

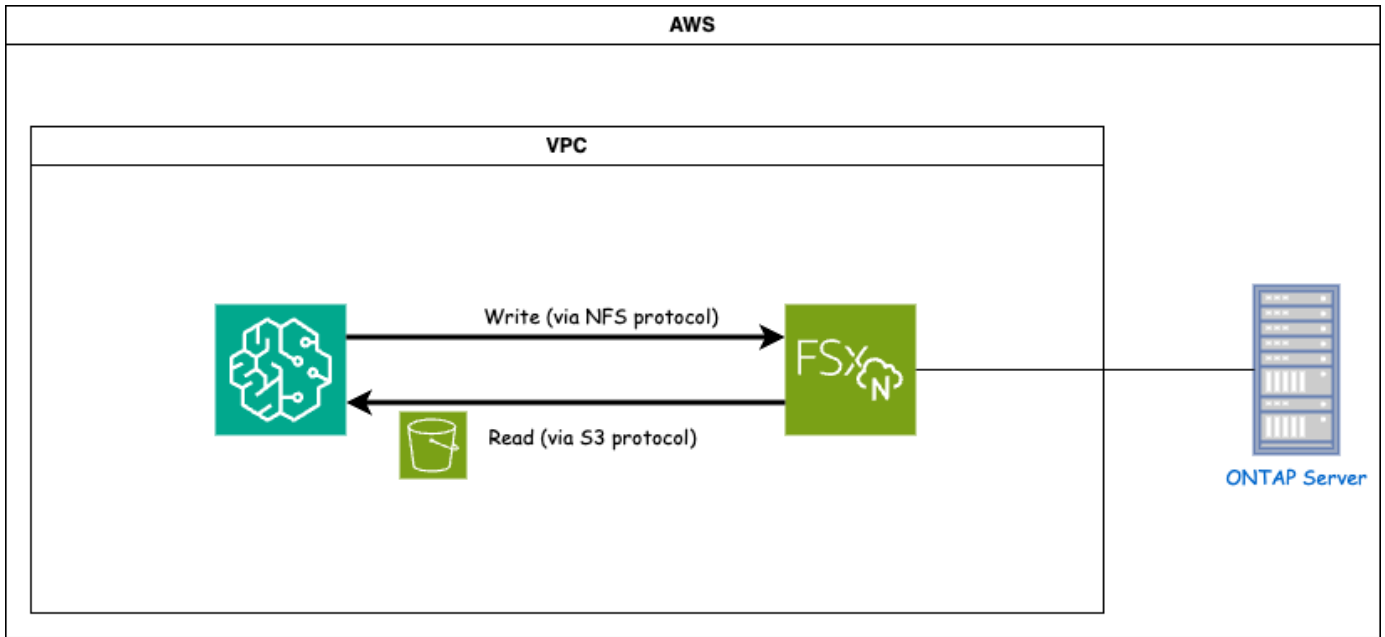
이 튜토리얼에서는 컴퓨터 비전 분류 프로젝트의 실제 사례를 제시하고, SageMaker 환경 내에서 FSx ONTAP 데이터 소스로 활용하는 ML 모델을 구축하는 실무 경험을 제공합니다. 이 프로젝트는 딥러닝 프레임워크인 PyTorch를 사용하여 타이어 이미지를 기반으로 타이어 품질을 분류하는 데 중점을 두고 있습니다. Amazon SageMaker에서 FSx ONTAP 데이터 소스로 사용하여 머신 러닝 모델을 개발하는 데 중점을 둡니다.

FSx ONTAP 무엇입니까?

Amazon FSx ONTAP AWS가 제공하는 완전 관리형 스토리지 솔루션입니다. NetApp의 ONTAP 파일 시스템을 활용하여 안정적이고 고성능의 스토리지를 제공합니다. NFS, SMB, iSCSI와 같은 프로토콜을 지원하므로 다양한 컴퓨팅 인스턴스와 컨테이너에서 원활하게 액세스할 수 있습니다. 이 서비스는 뛰어난 성능을 제공하고 빠르고 효율적인 데이터 작업을 보장하도록 설계되었습니다. 또한 높은 가용성과 내구성을 제공하여 데이터에 항상 접근하고 보호할 수 있습니다. 또한 Amazon FSx ONTAP의 저장 용량은 확장 가능하므로 필요에 따라 쉽게 조정할 수 있습니다.

필수 조건

네트워크 환경



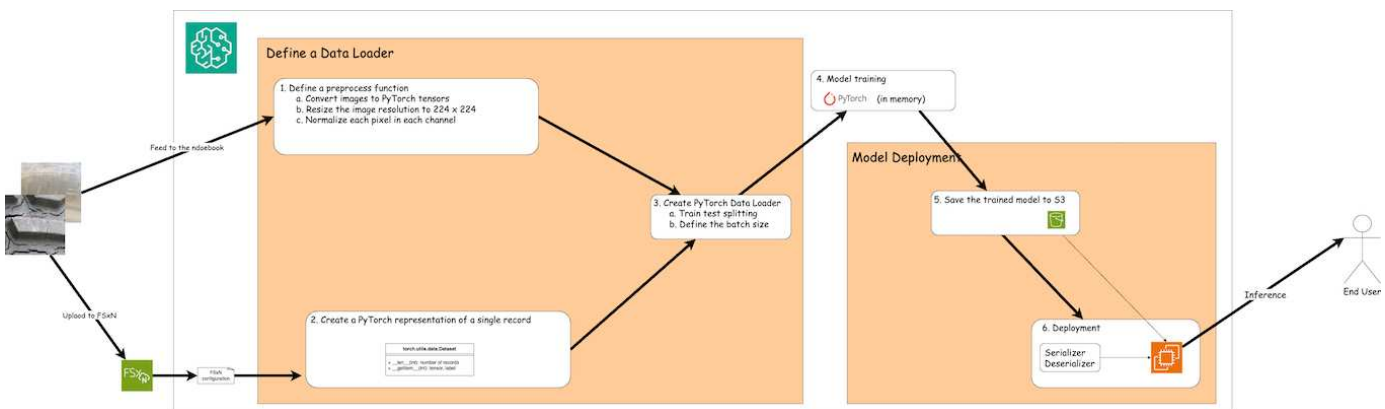
FSx ONTAP (Amazon FSx ONTAP)은 AWS 스토리지 서비스입니다. 여기에는 NetApp ONTAP 시스템에서 실행되는 파일 시스템과 이에 연결되는 AWS 관리 시스템 가상 머신(SVM)이 포함됩니다. 제공된 다이어그램에서 AWS가 관리하는 NetApp ONTAP 서버는 VPC 외부에 있습니다. SVM은 SageMaker와 NetApp ONTAP 시스템 사이의 중개자 역할을 하며, SageMaker로부터 작업 요청을 수신하여 기본 스토리지로 전달합니다. FSx ONTAP에 액세스하려면 SageMaker가 FSx ONTAP 배포와 동일한 VPC 내에 있어야 합니다. 이 구성은 SageMaker와 FSx ONTAP 간의 통신과 데이터 액세스를 보장합니다.

데이터 액세스

실제 상황에서 데이터 과학자는 일반적으로 FSx ONTAP에 저장된 기존 데이터를 활용하여 머신 러닝 모델을 구축합니다. 그러나 데모 목적으로, FSx ONTAP 파일 시스템은 생성된 후 처음에는 비어 있으므로 수동으로 교육 데이터를 업로드해야 합니다. 이는 FSx ONTAP SageMaker에 볼륨으로 마운트하여 달성할 수 있습니다. 파일 시스템이 성공적으로 마운트되면 마운트된 위치에 데이터 세트를 업로드하여 SageMaker 환경 내에서 모델을 학습할 수 있습니다. 이 접근 방식을 사용하면 SageMaker를 사용하여 모델을 개발하고 학습하는 동안 FSx ONTAP의 저장 용량과 기능을 활용할 수 있습니다.

데이터 읽기 프로세스에는 FSx ONTAP 개인 S3 버킷으로 구성하는 작업이 포함됩니다. 자세한 구성 지침을 알아보려면 다음을 참조하세요. ["1부 - Amazon FSx for NetApp ONTAP \(FSx ONTAP\)을 AWS SageMaker에 개인 S3 버킷으로 통합"](#)

통합 개요



FSx ONTAP의 학습 데이터를 사용하여 SageMaker에서 딥러닝 모델을 구축하는 워크플로는 데이터 로더 정의, 모델 학습, 배포의 세 가지 주요 단계로 요약할 수 있습니다. 높은 수준에서 이러한 단계는 MLOps 파이프라인의 기반을 형성합니다. 그러나 각 단계에는 포괄적인 구현을 위한 여러 가지 세부적인 하위 단계가 포함됩니다. 이러한 하위 단계에는 데이터 전처리, 데이터 세트 분할, 모델 구성, 하이퍼파라미터 튜닝, 모델 평가, 모델 배포와 같은 다양한 작업이 포함됩니다. 이러한 단계를 거치면 SageMaker 환경 내에서 FSx ONTAP의 학습 데이터를 사용하여 딥러닝 모델을 구축하고 배포하는 철저하고 효과적인 프로세스가 보장됩니다.

단계별 통합

데이터 Loader

PyTorch 딥러닝 네트워크를 데이터로 학습시키기 위해, 데이터 공급을 용이하게 하는 데이터 로더가 만들어졌습니다. 데이터 로더는 배치 크기를 정의할 뿐만 아니라 배치 내의 각 레코드를 읽고 사전 처리하는 절차도 결정합니다. 데이터 로더를 구성하면 일괄적으로 데이터를 처리하여 딥러닝 네트워크를 학습시킬 수 있습니다.

데이터 로더는 3개 부분으로 구성됩니다.

전처리 기능

```
from torchvision import transforms

preprocess = transforms.Compose([
    transforms.ToTensor(),
    transforms.Resize((224, 224)),
    transforms.Normalize(
        mean=[0.485, 0.456, 0.406],
        std=[0.229, 0.224, 0.225]
    )
])
```

위의 코드 조각은 **torchvision.transforms** 모듈을 사용하여 이미지 전처리 변환의 정의를 보여줍니다. 이 튜토리얼에서는 일련의 변환을 적용하기 위해 전처리 객체를 생성합니다. 첫째, **ToTensor()** 변환은 이미지를 텐서 표현으로 변환합니다. 이후, **Resize 224,224** 변환은 이미지 크기를 224x224픽셀의 고정 크기로 조정합니다. 마지막으로, **Normalize()** 변환은 각 채널의 평균을 빼고 표준 편차로 나누어 텐서 값을 정규화합니다. 정규화에 사용되는 평균 및 표준 편차 값은 일반적으로 사전 학습된 신경망 모델에 사용됩니다. 전반적으로 이 코드는 이미지 데이터를 텐서로 변환하고, 크기를 조정하고, 픽셀 값을 정규화하여 추가 처리나 사전 학습된 모델에 입력할 수 있도록 이미지 데이터를 준비합니다.

PyTorch 데이터셋 클래스

```

import torch
from io import BytesIO
from PIL import Image

class FSxNImageDataset(torch.utils.data.Dataset):
    def __init__(self, bucket, prefix='', preprocess=None):
        self.image_keys = [
            s3_obj.key
            for s3_obj in list(bucket.objects.filter(Prefix=prefix).all())
        ]
        self.preprocess = preprocess

    def __len__(self):
        return len(self.image_keys)

    def __getitem__(self, index):
        key = self.image_keys[index]
        response = bucket.Object(key)

        label = 1 if key[13:].startswith('defective') else 0

        image_bytes = response.get()['Body'].read()
        image = Image.open(BytesIO(image_bytes))
        if image.mode == 'L':
            image = image.convert('RGB')

        if self.preprocess is not None:
            image = self.preprocess(image)
        return image, label

```

이 클래스는 데이터 세트의 총 레코드 수를 얻는 기능을 제공하고 각 레코드에 대한 데이터를 읽는 방법을 정의합니다. **getitem** 함수 내에서 코드는 boto3 S3 버킷 객체를 활용하여 FSx ONTAP 에서 바이너리 데이터를 검색합니다. FSx ONTAP 에서 데이터에 액세스하는 코드 스타일은 Amazon S3에서 데이터를 읽는 것과 비슷합니다. 이후 설명에서는 개인 S3 객체 *버킷*의 생성 과정에 대해 자세히 설명합니다.

FSx ONTAP 개인 **S3** 저장소로 사용

```

seed = 77 # Random seed
bucket_name = '<Your ONTAP bucket name>' # The bucket
name in ONTAP
aws_access_key_id = '<Your ONTAP bucket key id>' # Please get
this credential from ONTAP
aws_secret_access_key = '<Your ONTAP bucket access key>' # Please get
this credential from ONTAP
fsx_endpoint_ip = '<Your FSx ONTAP IP address>' # Please
get this IP address from FSXN

```

```

import boto3

# Get session info
region_name = boto3.session.Session().region_name

# Initialize Fsx S3 bucket object
# --- Start integrating SageMaker with FSXN ---
# This is the only code change we need to incorporate SageMaker with FSXN
s3_client: boto3.client = boto3.resource(
    's3',
    region_name=region_name,
    aws_access_key_id=aws_access_key_id,
    aws_secret_access_key=aws_secret_access_key,
    use_ssl=False,
    endpoint_url=f'http://{fsx_endpoint_ip}',
    config=boto3.session.Config(
        signature_version='s3v4',
        s3={'addressing_style': 'path'}
    )
)
# s3_client = boto3.resource('s3')
bucket = s3_client.Bucket(bucket_name)
# --- End integrating SageMaker with FSXN ---

```

SageMaker에서 FSx ONTAP 에서 데이터를 읽으려면 S3 프로토콜을 사용하여 FSx ONTAP 저장소를 가리키는 핸들러가 생성됩니다. 이를 통해 FSx ONTAP 개인 S3 버킷으로 처리할 수 있습니다. 핸들러 구성에는 FSx ONTAP SVM의 IP 주소, 버킷 이름 및 필요한 자격 증명을 지정하는 것이 포함됩니다. 이러한 구성 항목을 얻는 것에 대한 포괄적인 설명은 다음 문서를 참조하십시오. "[1부 - Amazon FSx for NetApp ONTAP \(FSx ONTAP\)을 AWS SageMaker에 개인 S3 버킷으로 통합](#)".

위에서 언급한 예에서 버킷 객체는 PyTorch 데이터세트 객체를 인스턴스화하는 데 사용됩니다. 데이터 세트 객체에 대해서는 다음 섹션에서 자세히 설명하겠습니다.

```

from torch.utils.data import DataLoader
torch.manual_seed(seed)

# 1. Hyperparameters
batch_size = 64

# 2. Preparing for the dataset
dataset = FSxNImageDataset(bucket, 'dataset/tyre', preprocess=preprocess)

train, test = torch.utils.data.random_split(dataset, [1500, 356])

data_loader = DataLoader(dataset, batch_size=batch_size, shuffle=True)

```

제공된 예에서는 배치 크기가 64로 지정되어 각 배치에 64개의 레코드가 포함된다는 것을 나타냅니다. PyTorch **Dataset** 클래스, 전처리 함수, 학습 배치 크기를 결합하여 학습을 위한 데이터 로더를 얻습니다. 이 데이터 로더는 학습 단계 동안 데이터 세트를 일괄적으로 반복하는 과정을 용이하게 합니다.

모델 학습

```

from torch import nn

class TyreQualityClassifier(nn.Module):
    def __init__(self):
        super().__init__()
        self.model = nn.Sequential(
            nn.Conv2d(3, 32, (3, 3)),
            nn.ReLU(),
            nn.Conv2d(32, 32, (3, 3)),
            nn.ReLU(),
            nn.Conv2d(32, 64, (3, 3)),
            nn.ReLU(),
            nn.Flatten(),
            nn.Linear(64 * (224 - 6) * (224 - 6), 2)
        )
    def forward(self, x):
        return self.model(x)

```

```

import datetime

num_epochs = 2
device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

model = TyreQualityClassifier()
fn_loss = torch.nn.CrossEntropyLoss()
optimizer = torch.optim.Adam(model.parameters(), lr=1e-3)

model.to(device)
for epoch in range(num_epochs):
    for idx, (X, y) in enumerate(data_loader):
        X = X.to(device)
        y = y.to(device)

        y_hat = model(X)

        loss = fn_loss(y_hat, y)
        optimizer.zero_grad()
        loss.backward()
        optimizer.step()
        current_time = datetime.datetime.now().strftime("%Y-%m-%d
%H:%M:%S")
        print(f"Current Time: {current_time} - Epoch [{epoch+1}/
{num_epochs}]- Batch [{idx + 1}] - Loss: {loss}", end='\r')

```

이 코드는 표준 PyTorch 학습 과정을 구현합니다. 이는 합성곱 계층과 선형 계층을 사용하여 타이어 품질을 분류하는 *TyreQualityClassifier*라는 신경망 모델을 정의합니다. 학습 루프는 데이터 배치를 반복하고, 손실을 계산하고, 역전파와 최적화를 사용하여 모델의 매개변수를 업데이트합니다. 또한 모니터링 목적으로 현재 시간, 에포크, 배치 및 손실을 인쇄합니다.

모델 배포

전개

```

import io
import os
import tarfile
import sagemaker

# 1. Save the PyTorch model to memory
buffer_model = io.BytesIO()
traced_model = torch.jit.script(model)
torch.jit.save(traced_model, buffer_model)

# 2. Upload to AWS S3
sagemaker_session = sagemaker.Session()
bucket_name_default = sagemaker_session.default_bucket()
model_name = f'tyre_quality_classifier.pth'

# 2.1. Zip PyTorch model into tar.gz file
buffer_zip = io.BytesIO()
with tarfile.open(fileobj=buffer_zip, mode="w:gz") as tar:
    # Add PyTorch pt file
    file_name = os.path.basename(model_name)
    file_name_with_extension = os.path.splitext(file_name)[-1]
    tarinfo = tarfile.TarInfo(file_name_with_extension)
    tarinfo.size = len(buffer_model.getbuffer())
    buffer_model.seek(0)
    tar.addfile(tarinfo, buffer_model)

# 2.2. Upload the tar.gz file to S3 bucket
buffer_zip.seek(0)
boto3.resource('s3') \
    .Bucket(bucket_name_default) \
    .Object(f'pytorch/{model_name}.tar.gz') \
    .put(Body=buffer_zip.getvalue())

```

SageMaker는 배포를 위해 모델을 S3에 저장해야 하므로 이 코드는 PyTorch 모델을 *Amazon S3*에 저장합니다. 모델을 *Amazon S3*에 업로드하면 SageMaker에서 접근할 수 있게 되어 배포된 모델에 대한 배포와 추론이 가능해집니다.

```

import time
from sagemaker.pytorch import PyTorchModel
from sagemaker.predictor import Predictor
from sagemaker.serializers import IdentitySerializer
from sagemaker.deserializers import JSONDeserializer

class TyreQualitySerializer(IdentitySerializer):

```



```

CONTENT_TYPE = 'application/x-torch'

def serialize(self, data):
    transformed_image = preprocess(data)
    tensor_image = torch.Tensor(transformed_image)

    serialized_data = io.BytesIO()
    torch.save(tensor_image, serialized_data)
    serialized_data.seek(0)
    serialized_data = serialized_data.read()

    return serialized_data

class TyreQualityPredictor(Predictor):
    def __init__(self, endpoint_name, sagemaker_session):
        super().__init__(
            endpoint_name,
            sagemaker_session=sagemaker_session,
            serializer=TyreQualitySerializer(),
            deserializer=JSONDeserializer(),
        )

sagemaker_model = PyTorchModel(
    model_data=f's3://{bucket_name_default}/pytorch/{model_name}.tar.gz',
    role=sagemaker.get_execution_role(),
    framework_version='2.0.1',
    py_version='py310',
    predictor_cls=TyreQualityPredictor,
    entry_point='inference.py',
    source_dir='code',
)

timestamp = int(time.time())
pytorch_endpoint_name = '{}-{}-{}'.format('tyre-quality-classifier', 'pt',
timestamp)
sagemaker_predictor = sagemaker_model.deploy(
    initial_instance_count=1,
    instance_type='ml.p3.2xlarge',
    endpoint_name=pytorch_endpoint_name
)

```

이 코드는 SageMaker에서 PyTorch 모델을 배포하는 것을 용이하게 합니다. PyTorch 텐서로 입력 데이터를 전처리하고 직렬화하는 사용자 정의 직렬화기인 **TyreQualitySerializer***를 정의합니다. ***TyreQualityPredictor** 클래스는 정의된 직렬화기와 **JSONDeserializer***를 활용하는 사용자 정의 예측기입니다. 또한 이 코드는 모델의 **S3** 위치, **IAM** 역할, 프레임워크 버전, 추론 진입점을 지정하기 위해 ***PyTorchModel** 객체를 생성합니다. 이 코드는 타임스탬프를 생성하고 모델과 타임스탬프를 기반으로 엔드포인트 이름을 구성합니다. 마지막으로, 인스턴스 수,

인스턴스 유형, 생성된 엔드포인트 이름을 지정하여 배포 메서드를 사용하여 모델을 배포합니다. 이를 통해 PyTorch 모델을 배포하고 SageMaker에서 추론에 액세스할 수 있습니다.

추론

```
image_object = list(bucket.objects.filter('dataset/tyre'))[0].get()
image_bytes = image_object['Body'].read()

with Image.open(with Image.open(BytesIO(image_bytes)) as image:
    predicted_classes = sagemaker_predictor.predict(image)

print(predicted_classes)
```

이는 배포된 엔드포인트를 사용하여 추론을 수행하는 예입니다.

3부 - 단순화된 MLOps 파이프라인(CI/CT/CD) 구축

이 문서에서는 AWS 서비스를 사용하여 MLOps 파이프라인을 구축하는 방법에 대한 가이드를 제공하며, 자동화된 모델 재교육, 배포 및 비용 최적화에 중점을 둡니다.

소개

이 튜토리얼에서는 다양한 AWS 서비스를 활용하여 CI(지속적 통합), CT(지속적 학습), CD(지속적 배포)를 포함하는 간단한 MLOps 파이프라인을 구성하는 방법을 알아봅니다. 기존 DevOps 파이프라인과 달리 MLOps는 운영 주기를 완료하기 위해 추가적인 고려 사항이 필요합니다. 이 튜토리얼을 따라하면 CT를 MLOps 루프에 통합하여 모델의 지속적인 학습과 추론을 위한 원활한 배포에 대한 통찰력을 얻을 수 있습니다. 이 튜토리얼에서는 AWS 서비스를 활용하여 엔드투엔드 MLOps 파이프라인을 구축하는 과정을 안내합니다.

명백한

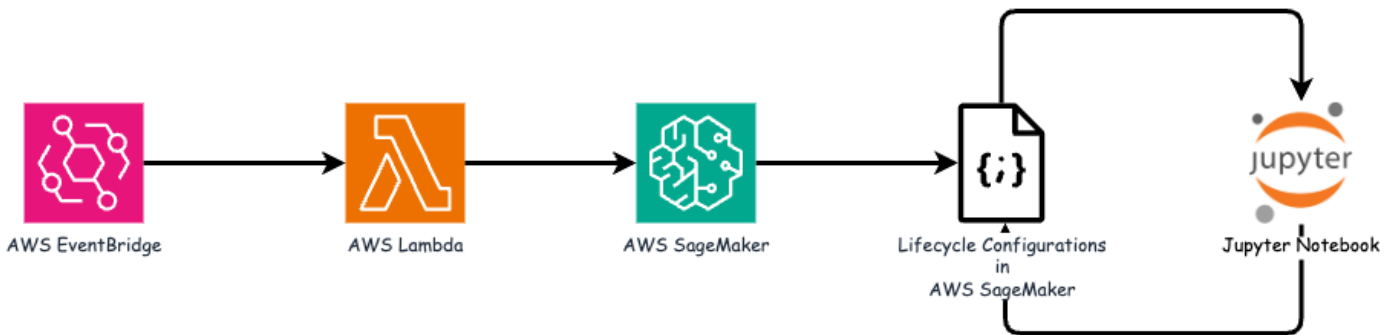
기능성	이름	논평
데이터 저장	AWS FSx ONTAP	"1부 - Amazon FSx for NetApp ONTAP (FSx ONTAP)을 AWS SageMaker에 개인 S3 버킷으로 통합"
데이터 과학 IDE	AWS 세이지메이커	이 튜토리얼은 Jupyter notebook을 기반으로 합니다."2부 - SageMaker에서 모델 학습을 위한 데이터 소스로 Amazon FSx for NetApp ONTAP (FSx ONTAP) 활용"
MLOps 파이프라인을 트리거하는 기능	AWS 람다 함수	-
Cron 작업 트리거	AWS 이벤트브리지	-
딥러닝 프레임워크	파이토치	-

기능성	이름	논평
AWS 파이썬 SDK	boto3	-
프로그래밍 언어	파이썬	v3.10

필수 조건

- 사전 구성된 FSx ONTAP 파일 시스템. 이 튜토리얼에서는 FSx ONTAP 에 저장된 데이터를 활용하여 학습 과정을 진행합니다.
- 위에 언급된 FSx ONTAP 파일 시스템과 동일한 VPC를 공유하도록 구성된 **SageMaker Notebook** 인스턴스.
- **AWS Lambda** 함수*를 트리거하기 전에 ***SageMaker Notebook** 인스턴스*가 ***중지 상태인지 확인**하세요.
- **ml.g4dn.xlarge** 인스턴스 유형은 딥 신경망의 계산에 필요한 GPU 가속을 활용하는 데 필요합니다.

아키텍처



이 MLOps 파이프라인은 Cron 작업을 활용하여 서버리스 함수를 트리거하고, 이를 통해 라이프사이클 콜백 함수에 등록된 AWS 서비스를 실행하는 실용적인 구현입니다. **AWS EventBridge***는 **cron** 작업 역할을 합니다. 주기적으로 모델을 재교육하고 재배포하는 **AWS Lambda** 함수를 호출합니다. 이 프로세스에는 필요한 작업을 수행하기 위해 ***AWS SageMaker Notebook** 인스턴스를 시작하는 작업이 포함됩니다.

단계별 구성

수명 주기 구성

AWS SageMaker Notebook 인스턴스에 대한 수명 주기 콜백 함수를 구성하려면 수명 주기 구성*을 활용합니다. 이 서비스를 사용하면 노트북 인스턴스를 시작할 때 수행해야 하는 필요한 작업을 정의할 수 있습니다. 구체적으로, 훈련 및 배포 프로세스가 완료되면 노트북 인스턴스를 자동으로 종료하기 위해 ***라이프사이클 구성** 내에 셀 스크립트를 구현할 수 있습니다. MLOps에서 비용은 주요 고려 사항 중 하나이므로 이는 필수 구성입니다.

***라이프사이클 구성**에 대한 구성은 미리 설정해야 한다는 점에 유의하세요. 따라서 다른 MLOps 파이프라인 설정을 진행하기 전에 이 측면의 구성을 우선시하는 것이 좋습니다.

1. 라이프사이클 구성을 설정하려면 **Sagemaker** 패널을 열고 관리자 구성 섹션 아래의 ***라이프사이클 구성***으로 이동합니다.

aws

Services

Search

S3

Amazon SageMaker

×

Getting started

Studio

Studio Lab

Canvas

RStudio

TensorBoard

Profiler

▼ Admin configurations

Domains

Role manager

Images

Lifecycle configurations

SageMaker dashboard

Search

► JumpStart

Amazon SageMaker > Domains

Domains

Info

A domain includes an associated Amazon SageMaker notebook instance. Each domain receives a personal and private Amazon SageMaker notebook instance.

► Domain structure diagram

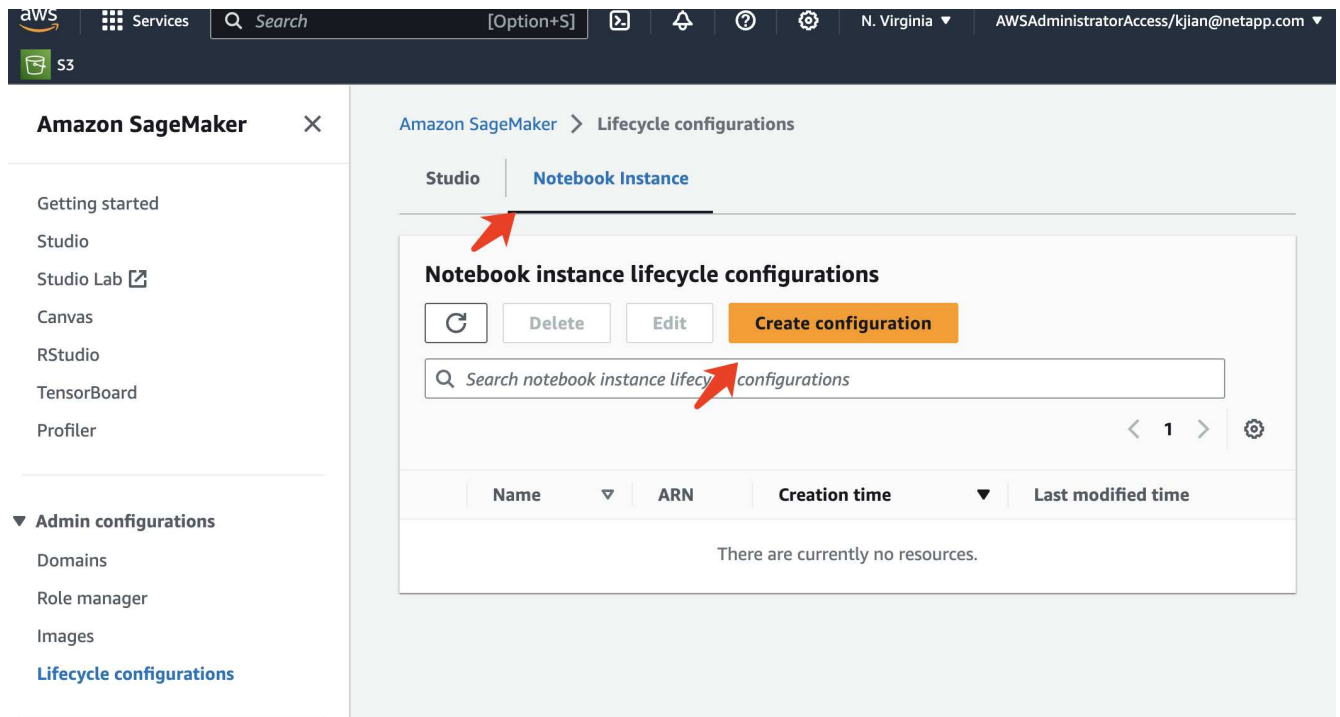
Domains (4)

Info

Find domain name

	Name
<input type="radio"/>	rdsml-east-1
<input type="radio"/>	rdsml-east-2
<input type="radio"/>	rdsml-east-3
<input type="radio"/>	rdsml-east-4

2. 노트북 인스턴스 탭을 선택하고 구성 생성 버튼을 클릭합니다.



3. 아래 코드를 입력란에 붙여넣으세요.

```
#!/bin/bash

set -e
sudo -u ec2-user -i <<'EOF'
# 1. Retraining and redeploying the model
NOTEBOOK_FILE=/home/ec2-user/SageMaker/tyre_quality_classification_local_training.ipynb
echo "Activating conda env"
source /home/ec2-user/anaconda3/bin/activate pytorch_p310
nohup jupyter nbconvert "$NOTEBOOK_FILE"
--ExecutePreprocessor.kernel_name=python --execute --to notebook &
nbconvert_pid=$!
conda deactivate

# 2. Scheduling a job to shutdown the notebook to save the cost
PYTHON_DIR='/home/ec2-user/anaconda3/envs/JupyterSystemEnv/bin/python3.10'
echo "Starting the autostop script in cron"
(crontab -l 2>/dev/null; echo "*/5 * * * * bash -c 'if ps -p
$nbconvert_pid > /dev/null; then echo \"Notebook is still running.\" >>
/var/log/jupyter.log; else echo \"Notebook execution completed.\" >>
/var/log/jupyter.log; $PYTHON_DIR -c \"import boto3;boto3.client(
\'sagemaker\').stop_notebook_instance(NotebookInstanceName=get_notebook_
name())\" >> /var/log/jupyter.log; fi')\" | crontab -
EOF
```

4. 이 스크립트는 추론을 위해 모델을 재교육하고 재배포하는 Jupyter Notebook을 실행합니다. 실행이 완료되면 5분 이내에 노트북이 자동으로 종료됩니다. 문제 설명과 코드 구현에 대해 자세히 알아보려면 다음을 참조하세요. ["2부 - SageMaker에서 모델 학습을 위한 데이터 소스로 Amazon FSx for NetApp ONTAP \(FSx ONTAP\) 활용"](#).

The screenshot shows the Amazon SageMaker console interface for creating a lifecycle configuration. The breadcrumb navigation is 'Amazon SageMaker > Lifecycle configurations > Create lifecycle configuration'. The main heading is 'Create lifecycle configuration'.

Configuration setting

Name:

Alphanumeric characters and "-", no spaces. Maximum 63 characters.

Scripts

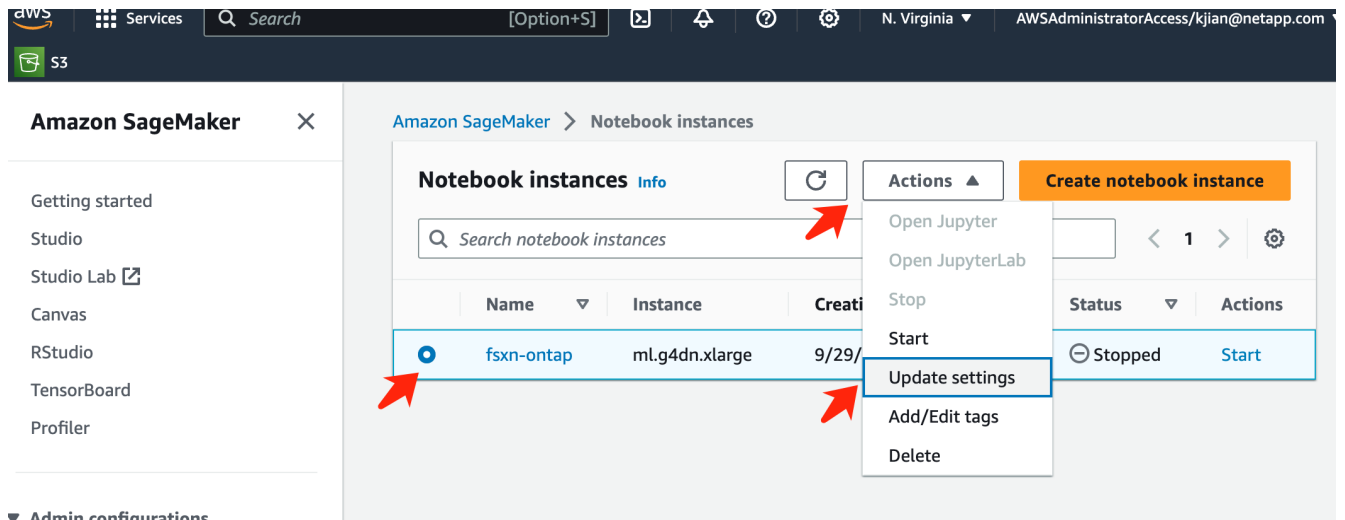
Start notebook | Create notebook

This script will be run each time an associated notebook instance is started, including during initial creation. If the associated notebook instance is already started, it will be run the next time it is stopped and started. [a curated list of sample scripts](#)

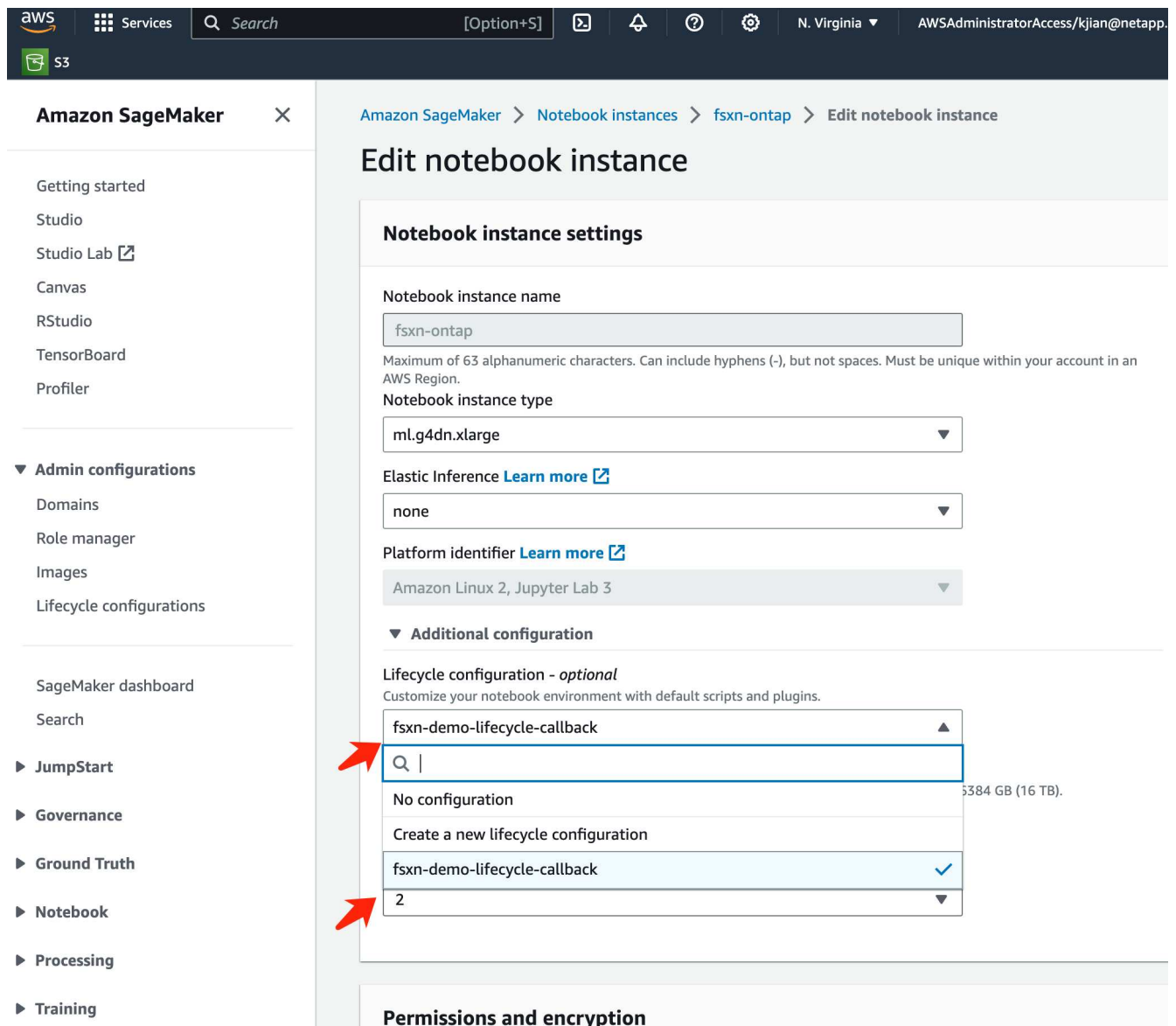
```
1 #!/bin/bash
2
3 set -e
4 sudo -u ec2-user -i <<'EOF'
5 # 1. Retraining and redeploying the model
6 NOTEBOOK_FILE=/home/ec2-user/SageMaker/tyre_quality_classification_local_training.ipynb
7 echo "Activating conda env"
8 source /home/ec2-user/anaconda3/bin/activate pytorch_p310
9 nohup jupyter nbconvert "$NOTEBOOK_FILE" --ExecutePreprocessor.kernel_name=python --execute --to n
10 nbconvert_pid=$!
11 conda deactivate
12
13 # 2. Scheduling a job to shutdown the notebook to save the cost
14 PYTHON_DIR="/home/ec2-user/anaconda3/envs/JupyterSystemEnv/bin/python3.10"
15 echo "Starting the autostop script in cron"
16 (crontab -l 2>/dev/null; echo "*/5 * * * * bash -c 'if ps -p $nbconvert_pid > /dev/null; then echo
17 EOF
```

Cancel **Create configuration**

5. 생성 후 Notebook 인스턴스로 이동하여 대상 인스턴스를 선택하고 작업 드롭다운에서 *설정 업데이트*를 클릭합니다.



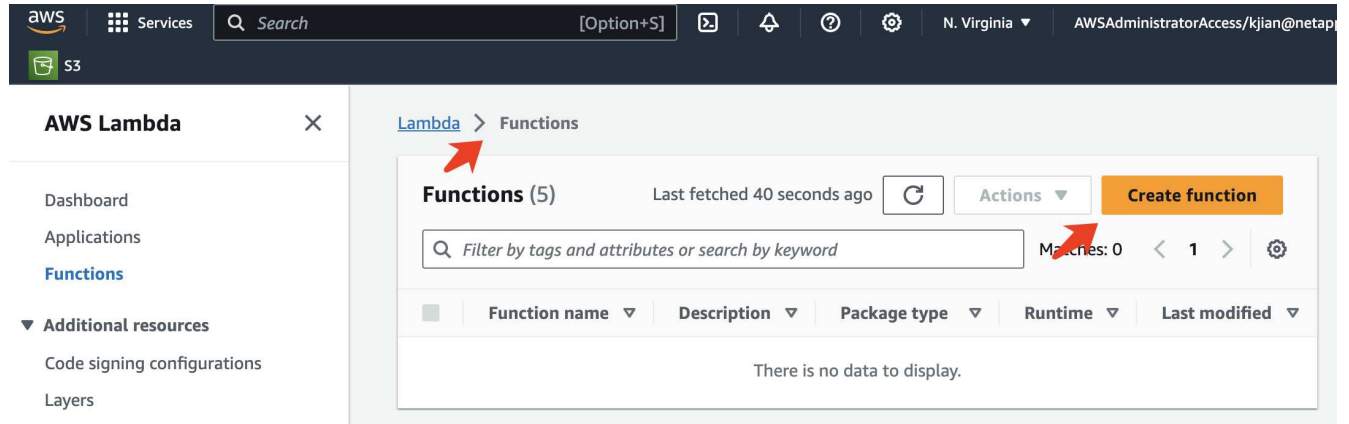
6. 생성된 *라이프사이클 구성*을 선택하고 *노트북 인스턴스 업데이트*를 클릭합니다.



AWS Lambda 서버리스 함수

앞서 언급했듯이 *AWS Lambda 함수*는 *AWS SageMaker Notebook 인스턴스*를 시작하는 역할을 합니다.

1. **AWS Lambda** 함수*를 만들려면 해당 패널로 이동하여 *함수 탭으로 전환하고 *함수 만들기*를 클릭합니다.



2. 모든 필수 항목을 페이지에 입력하시고 런타임을 *Python 3.10*으로 전환하는 것을 잊지 마세요.

aws Services Search [Option+S] N. Virgi AWSAdministratorAccess/kjian@

S3

Lambda > Functions > Create function

Create function [Info](#)

AWS Serverless Application Repository applications have moved to [Create application](#).

☒ **Author from scratch**
Start with a simple Hello World example.

☐ **Use a blueprint**
Build a Lambda application from sample code and configuration presets for common use cases.

☐ **Container image**
Select a container image to deploy for your function.

Basic information

Function name
Enter a name that describes the purpose of your function.

fsxn-demo-mlops

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.10

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.

☒ x86_64

☐ arm64

Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

3. 지정된 역할에 필요한 권한인 **AmazonSageMakerFullAccess***가 있는지 확인하고 *함수 생성 버튼을 클릭하세요.

aws Services Search [Option+S] N. Virgi AWSAdministratorAccess/kjian@

S3

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.10

Architecture [Info](#)
Choose the instruction set architecture you want for your function code.
☒ x86_64
☐ arm64

Permissions [Info](#)
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ **Change default execution role**

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

☐ Create a new role with basic Lambda permissions
☒ Use an existing role
☐ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
service-role/fsxn-demo-mlops-role-585jzdny
[View the fsxn-demo-mlops-role-585jzdny role](#) on the IAM console.

► **Advanced settings**

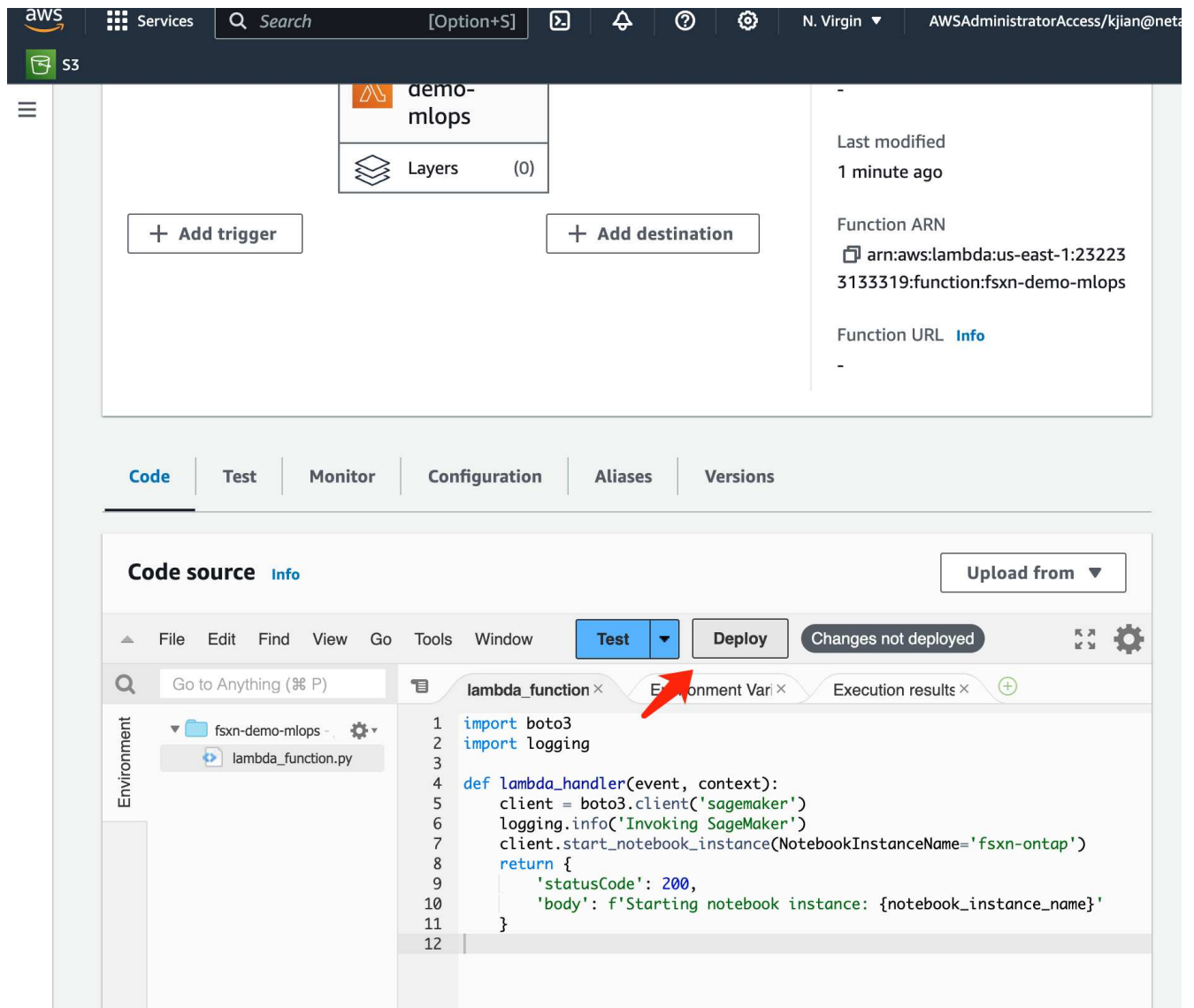
Cancel Create function

4. 생성된 Lambda 함수를 선택합니다. 코드 탭에서 다음 코드를 복사하여 텍스트 영역에 붙여넣습니다. 이 코드는 *fsxn-ontap*이라는 노트북 인스턴스를 시작합니다.

```
import boto3
import logging

def lambda_handler(event, context):
    client = boto3.client('sagemaker')
    logging.info('Invoking SageMaker')
    client.start_notebook_instance(NotebookInstanceName='fsxn-ontap')
    return {
        'statusCode': 200,
        'body': f'Starting notebook instance: {notebook_instance_name}'
    }
```

5. 이 코드 변경 사항을 적용하려면 배포 버튼을 클릭하세요.



6. 이 AWS Lambda 함수를 트리거하는 방법을 지정하려면 트리거 추가 버튼을 클릭합니다.

The screenshot shows the AWS Lambda console interface. At the top, the navigation bar includes the AWS logo, 'Services', a search bar, and the user's profile. The breadcrumb trail indicates the path: [Lambda](#) > [Functions](#) > fsxn-demo-mlops. The function name 'fsxn-demo-mlops' is prominently displayed. To the right of the name are buttons for 'Throttle', 'Copy ARN', and 'Actions'. Below the function name, the 'Function overview' section is expanded, showing a summary of the function. A red arrow points to the '+ Add trigger' button, which is located on the left side of the overview panel. The overview panel also displays the function's icon, name, layers, and a list of metadata on the right, including the function ARN and last modified time.

aws Services Search [Option+S] N. Virginia AWSAdministratorAccess/kjian@netapp.

S3

[Lambda](#) > [Functions](#) > fsxn-demo-mlops

fsxn-demo-mlops

Throttle Copy ARN Actions

▼ Function overview Info

fsxn-demo-mlops

Layers (0)

+ Add trigger

+ Add destination

Description -

Last modified 2 minutes ago

Function ARN
arn:aws:lambda:us-east-1:232233133319:function:fsxn-demo-mlops

Function URL Info

7. 드롭다운 메뉴에서 EventBridge를 선택한 다음, 새 규칙 만들기라는 라벨이 있는 라디오 버튼을 클릭합니다. 일정 표현 필드에 다음을 입력하세요. `rate(1 day)` , 추가 버튼을 클릭하여 이 새로운 Cron 작업 규칙을 생성하고 AWS Lambda 함수에 적용합니다.

aws Services Search [Option+S] N. Virginia AWSAdministratorAccess

S3

[Lambda](#) > Add trigger

Add trigger

Trigger configuration [Info](#)

EventBridge (CloudWatch Events)
aws asynchronous schedule management-tools

Rule
Pick an existing rule, or create a new one.

☒ Create a new rule
☐ Existing rules

Rule name
Enter a name to uniquely identify your rule.

mlops-retraining-trigger

Rule description
Provide an optional description for your rule.

Rule type
Trigger your target based on an event pattern, or based on an automated schedule.

☐ Event pattern
☒ Schedule expression

Schedule expression
Self-trigger your target on an automated schedule using [Cron or rate expressions](#). Cron expressions are in UTC.

rate(1 day)

e.g. rate(1 day), cron(0 17 ? * MON-FRI *)

Lambda will add the necessary permissions for Amazon EventBridge (CloudWatch Events) to invoke your Lambda function from this trigger. [Learn more](#) about the Lambda permissions model.

Cancel Add

2단계 구성을 완료한 후에는 매일 **AWS Lambda** 함수가 ***SageMaker Notebook***을 시작하고, ***FSx ONTAP*** 저장소의 데이터를 사용하여 모델 재교육을 수행하고, 업데이트된 모델을 프로덕션 환경으로 다시 배포하고, 비용을 최적화하기 위해 ***SageMaker Notebook*** 인스턴스를 자동으로 종료합니다. 이렇게 하면 모델이 최신 상태로 유지됩니다.

이것으로 MLOps 파이프라인을 개발하기 위한 튜토리얼이 끝났습니다.

저작권 정보

Copyright © 2026 NetApp, Inc. All Rights Reserved. 미국에서 인쇄된 본 문서의 어떠한 부분도 저작권 소유자의 사전 서면 승인 없이는 어떠한 형식이나 수단(복사, 녹음, 녹화 또는 전자 검색 시스템에 저장하는 것을 비롯한 그래픽, 전자적 또는 기계적 방법)으로도 복제될 수 없습니다.

NetApp이 저작권을 가진 자료에 있는 소프트웨어에는 아래의 라이선스와 고지사항이 적용됩니다.

본 소프트웨어는 NetApp에 의해 '있는 그대로' 제공되며 상품성 및 특정 목적에의 적합성에 대한 명시적 또는 묵시적 보증을 포함하여(이에 제한되지 않음) 어떠한 보증도 하지 않습니다. NetApp은 대체품 또는 대체 서비스의 조달, 사용 불능, 데이터 손실, 이익 손실, 영업 중단을 포함하여(이에 국한되지 않음), 이 소프트웨어의 사용으로 인해 발생하는 모든 직접 및 간접 손해, 우발적 손해, 특별 손해, 징벌적 손해, 결과적 손해의 발생에 대하여 그 발생 이유, 책임론, 계약 여부, 엄격한 책임, 불법 행위(과실 또는 그렇지 않은 경우)와 관계없이 어떠한 책임도 지지 않으며, 이와 같은 손실의 발생 가능성이 통지되었다 하더라도 마찬가지입니다.

NetApp은 본 문서에 설명된 제품을 언제든지 예고 없이 변경할 권리를 보유합니다. NetApp은 NetApp의 명시적인 서면 동의를 받은 경우를 제외하고 본 문서에 설명된 제품을 사용하여 발생하는 어떠한 문제에도 책임을 지지 않습니다. 본 제품의 사용 또는 구매의 경우 NetApp에서는 어떠한 특허권, 상표권 또는 기타 지적 재산권이 적용되는 라이선스도 제공하지 않습니다.

본 설명서에 설명된 제품은 하나 이상의 미국 특허, 해외 특허 또는 출원 중인 특허로 보호됩니다.

제한적 권리 표시: 정부에 의한 사용, 복제 또는 공개에는 DFARS 252.227-7013(2014년 2월) 및 FAR 52.227-19(2007년 12월)의 기술 데이터-비상업적 품목에 대한 권리(Rights in Technical Data -Noncommercial Items) 조항의 하위 조항 (b)(3)에 설명된 제한사항이 적용됩니다.

여기에 포함된 데이터는 상업용 제품 및/또는 상업용 서비스(FAR 2.101에 정의)에 해당하며 NetApp, Inc.의 독점 자산입니다. 본 계약에 따라 제공되는 모든 NetApp 기술 데이터 및 컴퓨터 소프트웨어는 본질적으로 상업용이며 개인 비용만으로 개발되었습니다. 미국 정부는 데이터가 제공된 미국 계약과 관련하여 해당 계약을 지원하는 데에만 데이터에 대한 전 세계적으로 비독점적이고 양도할 수 없으며 재사용이 불가능하며 취소 불가능한 라이선스를 제한적으로 가집니다. 여기에 제공된 경우를 제외하고 NetApp, Inc.의 사전 서면 승인 없이는 이 데이터를 사용, 공개, 재생산, 수정, 수행 또는 표시할 수 없습니다. 미국 국방부에 대한 정부 라이선스는 DFARS 조항 252.227-7015(b)(2014년 2월)에 명시된 권한으로 제한됩니다.

상표 정보

NETAPP, NETAPP 로고 및 <http://www.netapp.com/TM>에 나열된 마크는 NetApp, Inc.의 상표입니다. 기타 회사 및 제품 이름은 해당 소유자의 상표일 수 있습니다.