



FSx ONTAP per MLOps

NetApp artificial intelligence solutions

NetApp

February 12, 2026

This PDF was generated from <https://docs.netapp.com/it-it/netapp-solutions-ai/cloud/ai-mlops-fsxn-intro.html> on February 12, 2026. Always check docs.netapp.com for the latest.

Sommario

- FSx ONTAP per MLOps 1
 - Amazon FSx for NetApp ONTAP (FSx ONTAP) per MLOps 1
- Parte 1 - Integrazione di Amazon FSx for NetApp ONTAP (FSx ONTAP) come bucket S3 privato in AWS SageMaker 1
 - Introduzione 1
 - Guida per l'utente 1
 - Lista di controllo utile per il debug 14
 - FAQ (aggiornato al 27 settembre 2023). 15
- Parte 2 - Utilizzo di AWS Amazon FSx for NetApp ONTAP (FSx ONTAP) come origine dati per l'addestramento del modello in SageMaker 15
 - Introduzione 15
 - Che cos'è FSx ONTAP 15
 - Prerequisito 16
 - Panoramica sull'integrazione 16
 - Integrazione passo dopo passo 17
- Parte 3 - Creazione di una pipeline MLOps semplificata (CI/CT/CD) 24
 - Introduzione 24
 - Manifesto 24
 - Prerequisito 25
 - Architettura 25
 - Configurazione passo passo 25

FSx ONTAP per MLOps

Amazon FSx for NetApp ONTAP (FSx ONTAP) per MLOps

Questa sezione approfondisce l'applicazione pratica dello sviluppo dell'infrastruttura AI, fornendo una guida completa alla costruzione di una pipeline MLOps utilizzando FSx ONTAP. Composto da tre esempi esaustivi, ti guida a soddisfare le tue esigenze MLOps tramite questa potente piattaforma di gestione dei dati.

Questi articoli si concentrano su:

1. ["Parte 1 - Integrazione di Amazon FSx for NetApp ONTAP \(FSx ONTAP\) come bucket S3 privato in AWS SageMaker"](#)
2. ["Parte 2 - Utilizzo di Amazon FSx for NetApp ONTAP \(FSx ONTAP\) come origine dati per l'addestramento del modello in SageMaker"](#)
3. ["Parte 3 - Creazione di una pipeline MLOps semplificata \(CI/CT/CD\)"](#)

Al termine di questa sezione, avrai acquisito una solida comprensione di come utilizzare FSx ONTAP per semplificare i processi MLOps.

Parte 1 - Integrazione di Amazon FSx for NetApp ONTAP (FSx ONTAP) come bucket S3 privato in AWS SageMaker

Questa sezione fornisce una guida sulla configurazione di FSx ONTAP come bucket S3 privato utilizzando AWS SageMaker.

Introduzione

Utilizzando SageMaker come esempio, questa pagina fornisce indicazioni sulla configurazione di FSx ONTAP come bucket S3 privato.

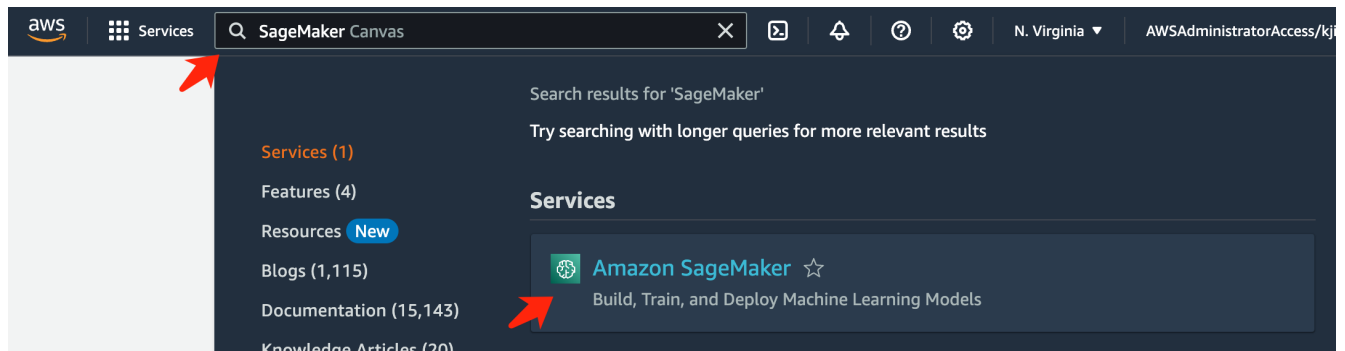
Per maggiori informazioni su FSx ONTAP, dai un'occhiata a questa presentazione (["Collegamento video"](#))

Guida per l'utente

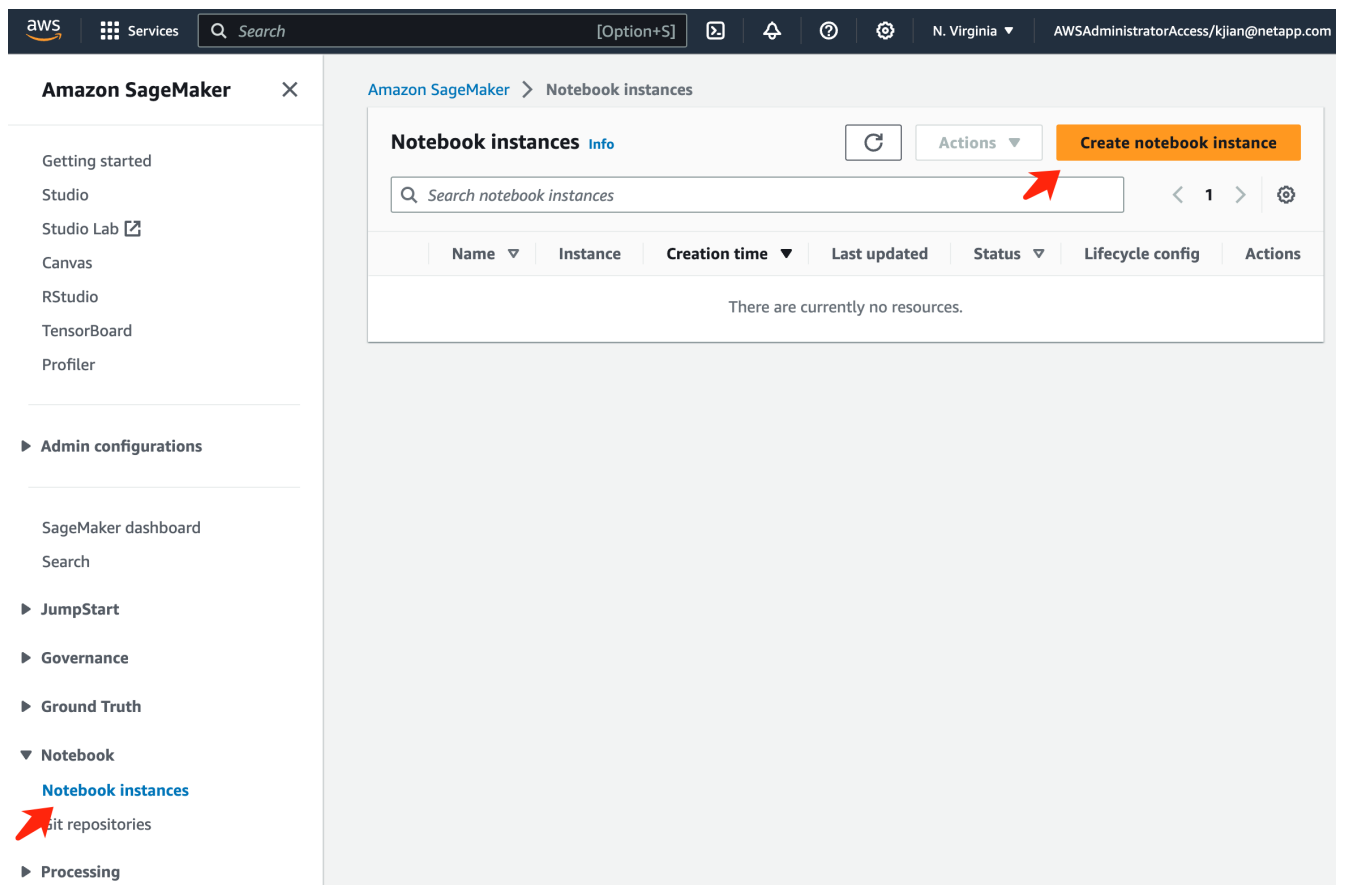
Creazione del server

Creare un'istanza di SageMaker Notebook

1. Apri la console AWS. Nel pannello di ricerca, cerca SageMaker e clicca sul servizio **Amazon SageMaker**.



2. Aprire le **Istanze del notebook** nella scheda Notebook, fare clic sul pulsante arancione **Crea istanza del notebook**.



3. Nella pagina di creazione, inserisci il **Nome dell'istanza del notebook**. Espandi il pannello **Rete**. Lascia le altre voci predefinite e seleziona una **VPC**, una **Subnet** e un **Gruppo/i di sicurezza**. (Questa **VPC** e **Subnet** verranno utilizzate in seguito per creare il file system FSx ONTAP) Fare clic sul pulsante arancione **Crea istanza notebook** in basso a destra.

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](#)

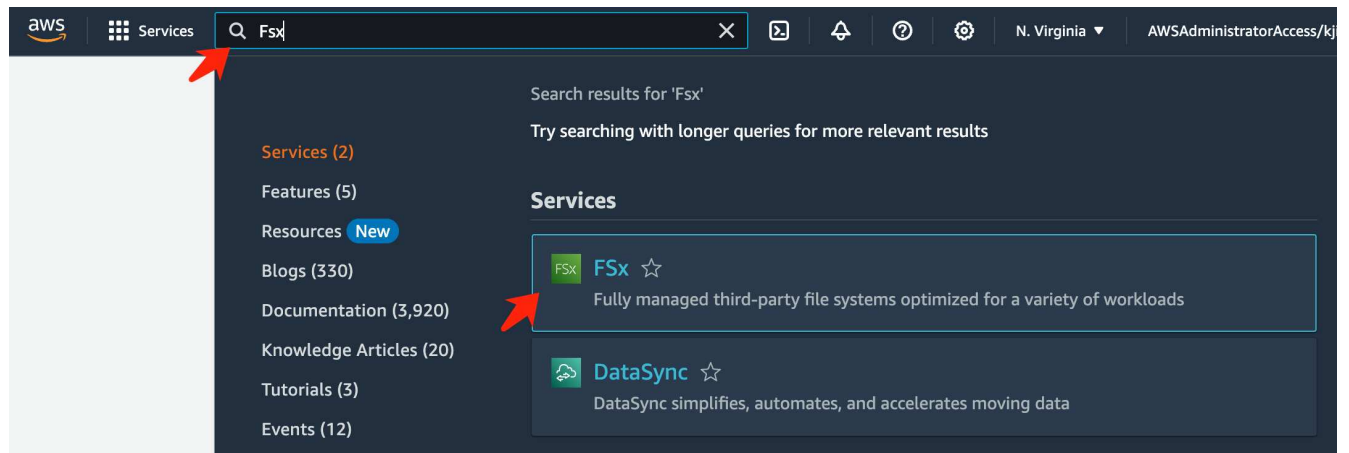
► Git repositories - optional

► Tags - optional

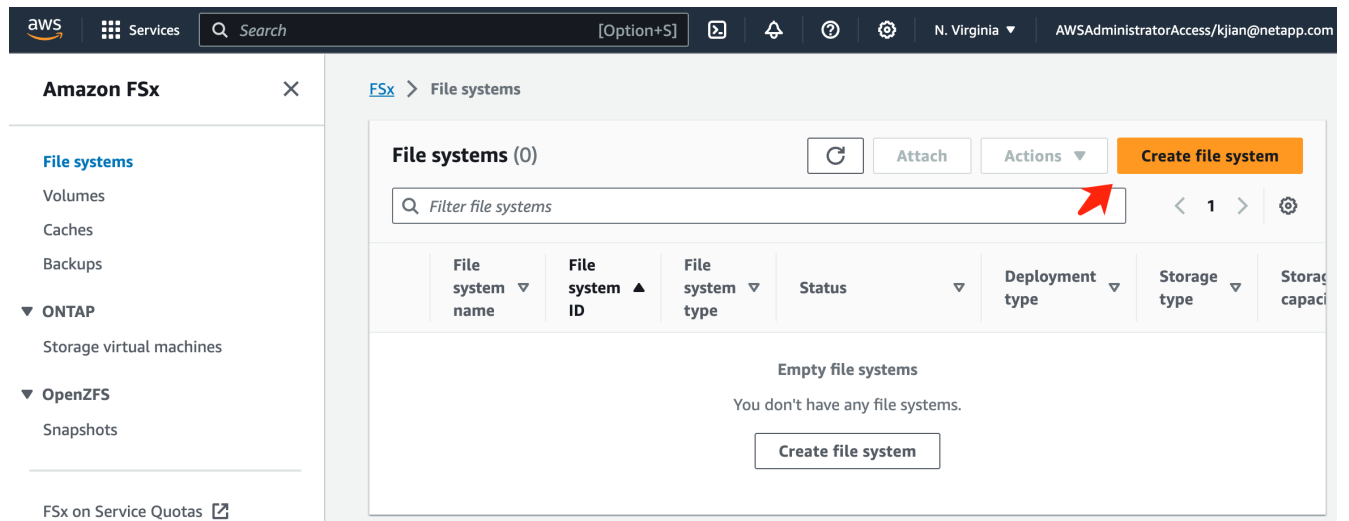
Cancel Create notebook instance

Creare un file system FSx ONTAP

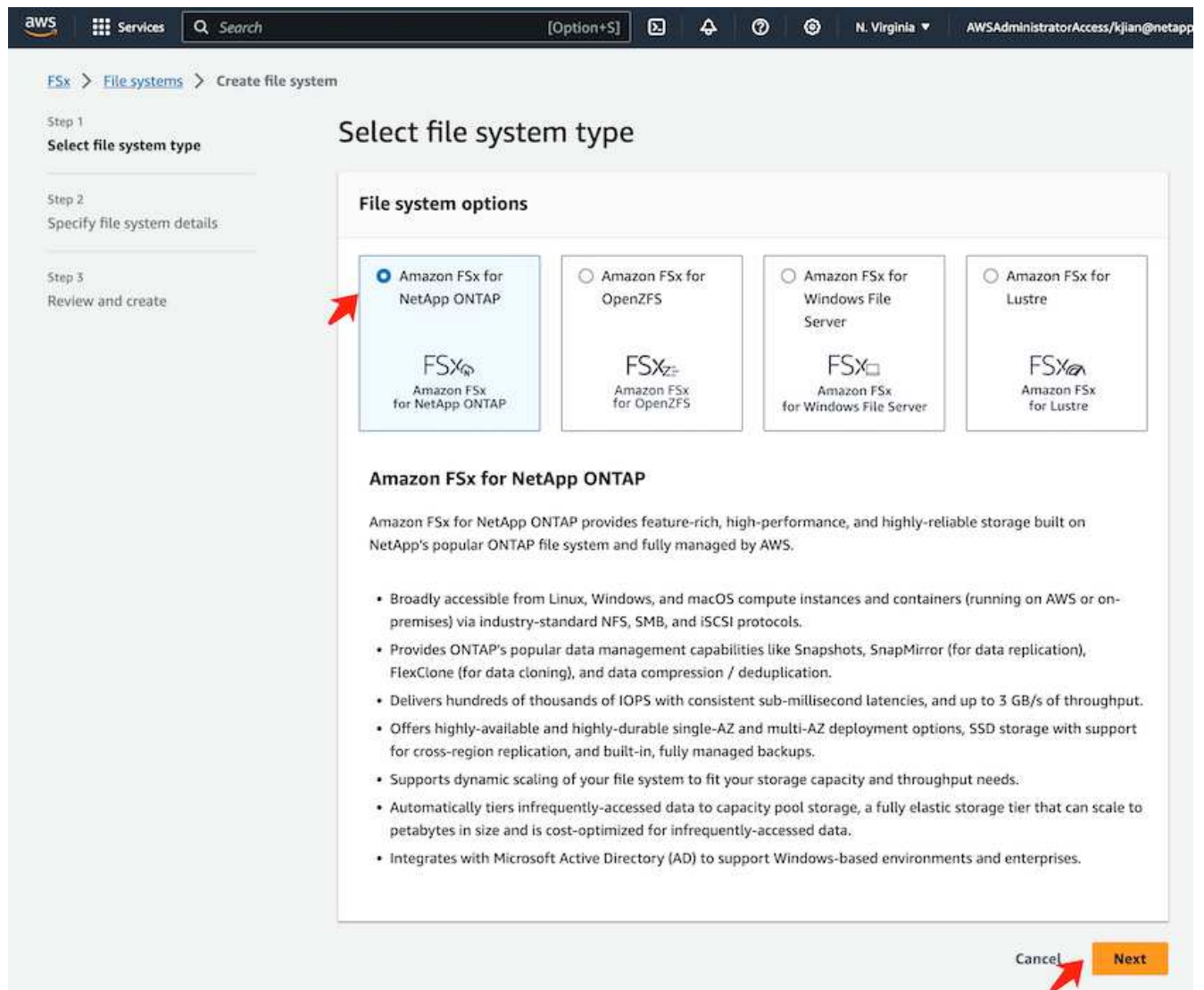
1. Apri la console AWS. Nel pannello di ricerca, cerca FSx e clicca sul servizio **FSx**.



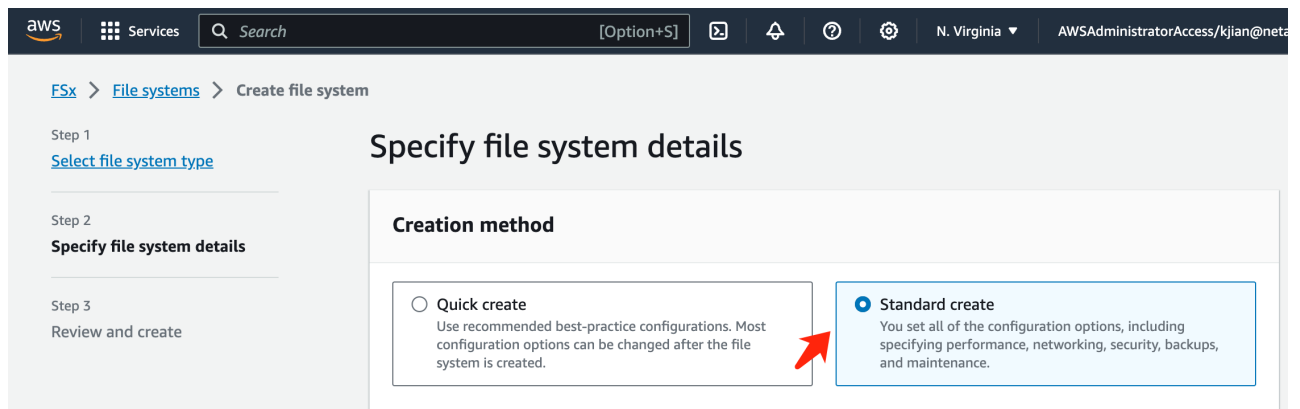
2. Fare clic su **Crea file system**.



3. Selezionare la prima scheda **FSx ONTAP** e fare clic su **Avanti**.



4. Nella pagina di configurazione dei dettagli.
- a. Selezionare l'opzione **Creazione standard**.



- b. Immettere il **Nome del file system** e la **Capacità di archiviazione SSD**.

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. Assicurarsi di utilizzare la **VPC** e la **subnet** identiche per l'istanza di **SageMaker Notebook**.

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. Inserisci il nome della **macchina virtuale di archiviazione** e **specifica una password** per la tua SVM (macchina virtuale di archiviazione).

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. Lascia le altre voci predefinite e clicca sul pulsante arancione **Avanti** in basso a destra.

► Backup and maintenance - optional

► Tags - optional

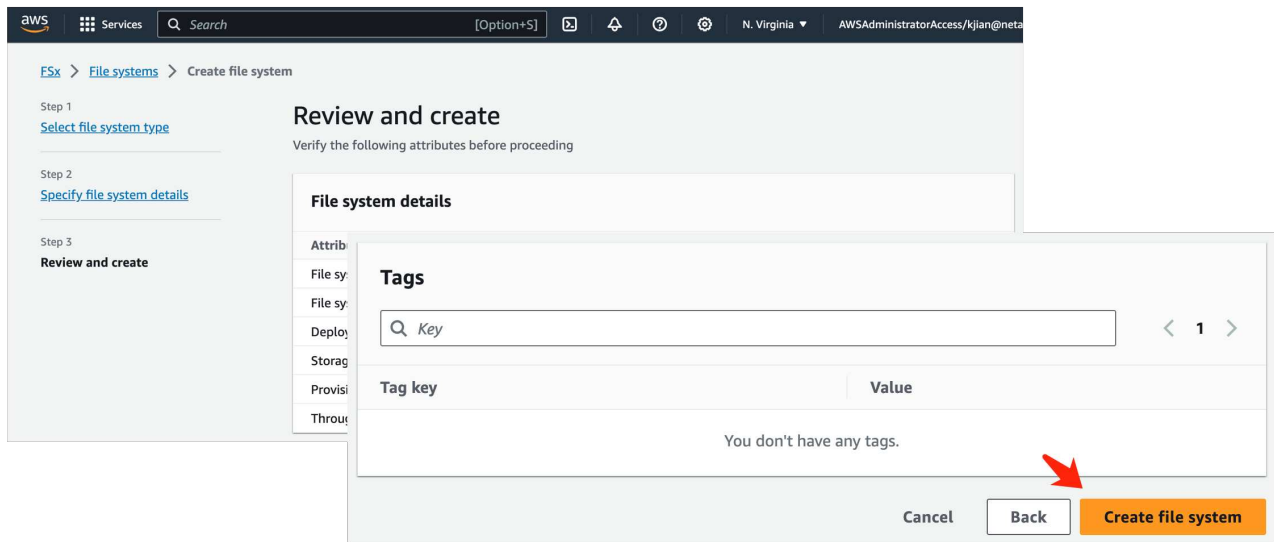
Cancel

Back

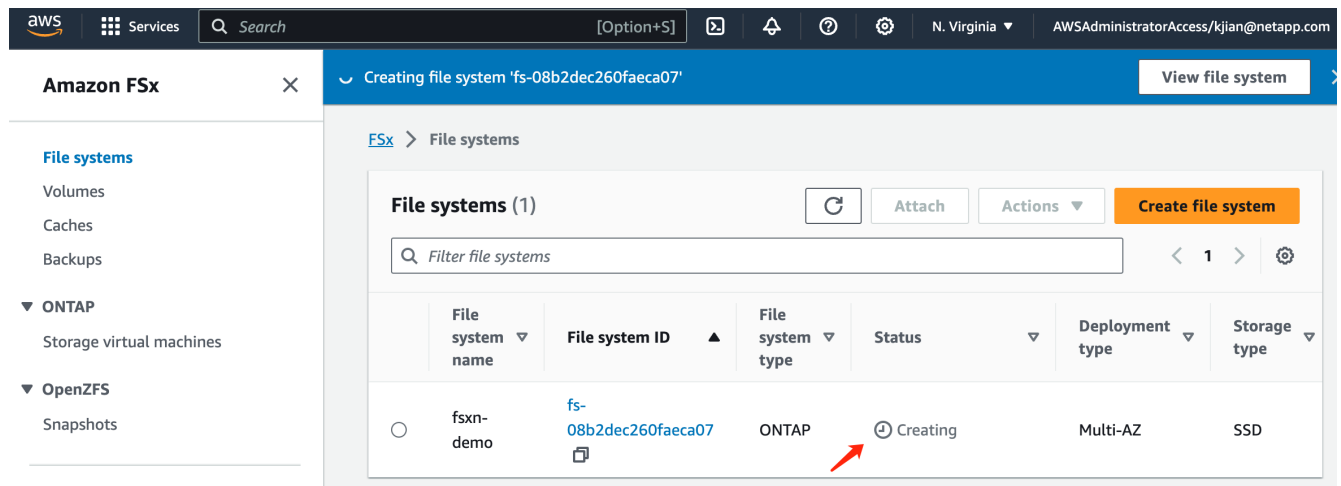
Next

f. Fare clic sul pulsante arancione **Crea file system** in basso a destra nella pagina di revisione.

8



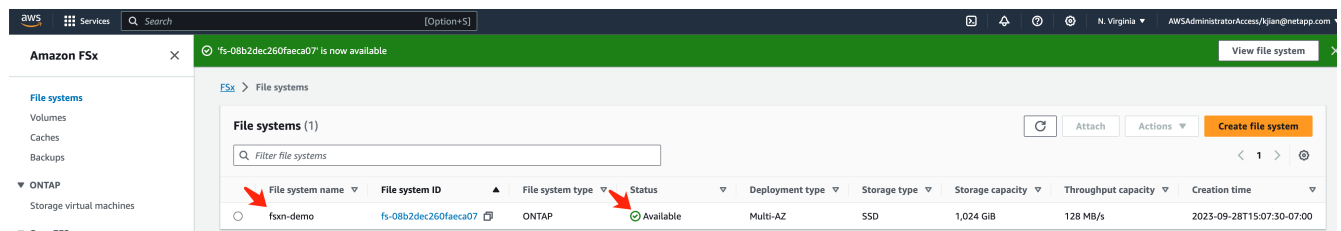
5. Potrebbero essere necessari circa **20-40 minuti** per avviare il file system FSx.



Configurazione del server

Configurazione ONTAP

1. Aprire il file system FSx creato. Assicurati che lo stato sia **Disponibile**.



2. Selezionare la scheda **Amministrazione** e mantenere **Endpoint di gestione - Indirizzo IP** e ***Nome utente amministratore ONTAP ***.

Amazon FSx ×

File systems
Volumes
Caches
Backups

▼ **ONTAP**
Storage virtual machines

▼ **OpenZFS**
Snapshots

FSx on Service Quotas [↗](#)

[FSx](#) > [File systems](#) > fs-08b2dec260faeca07

fsxn-demo (fs-08b2dec260faeca07) Attach Actions ▼

▼ **Summary**

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

< Network & security Monitoring & performance **Administration** Storage virtual machines >

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 ↗ 172.31.32.38 ↗	ONTAP administrator password Update

3. Aprire l'istanza **SageMaker Notebook** creata e fare clic su **Apri JupyterLab**.

Amazon SageMaker ×

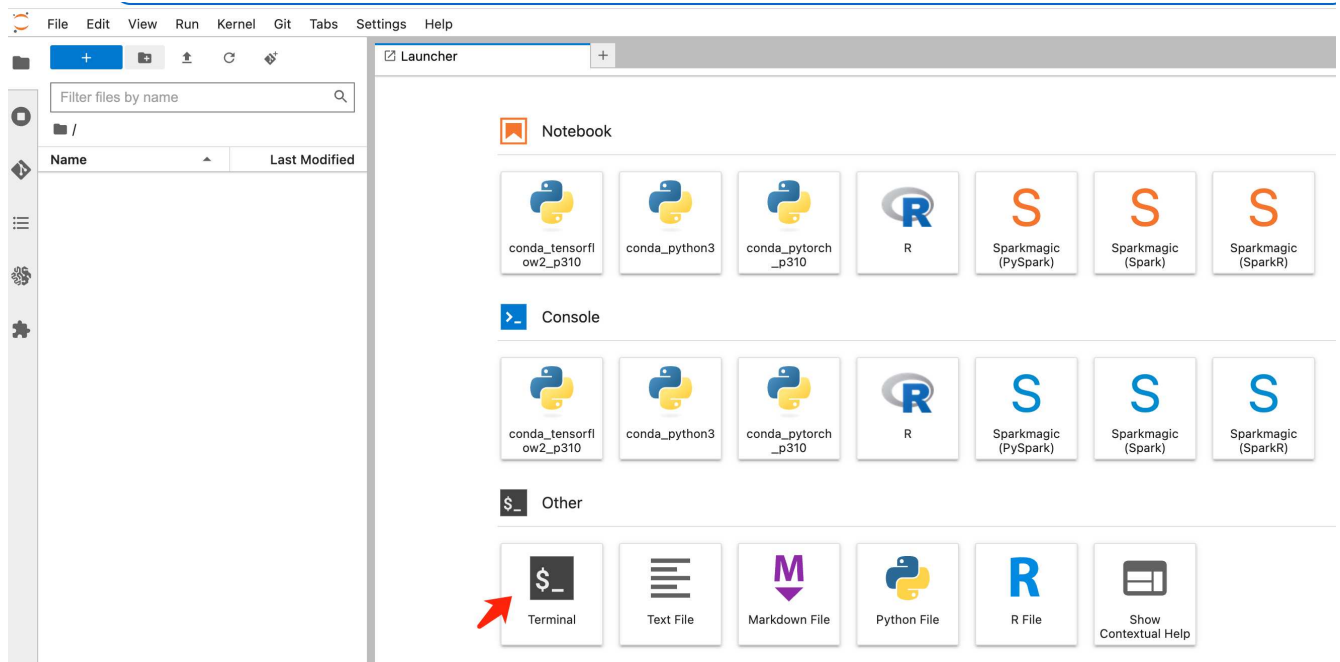
Getting started
Studio
Studio Lab [↗](#)
Canvas
RStudio
TensorBoard

[Amazon SageMaker](#) > Notebook instances

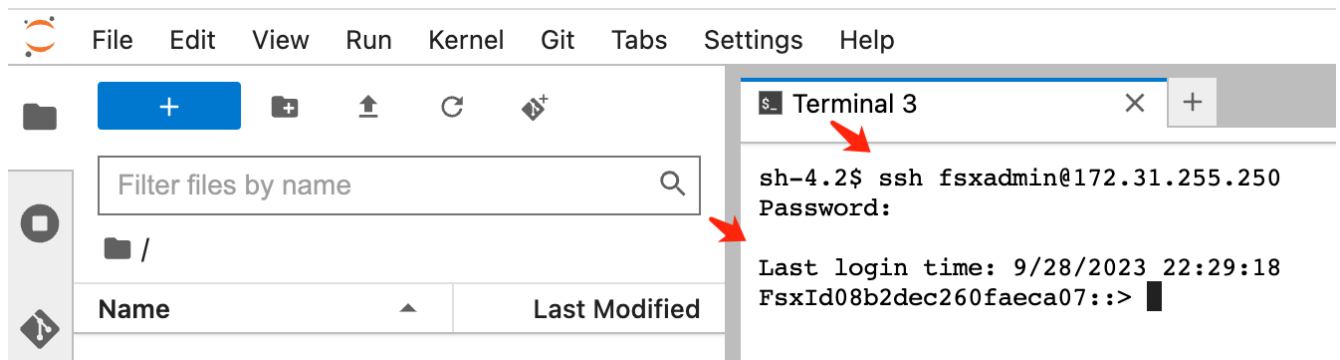
Notebook instances [Info](#) ↻ Actions ▼ Create notebook instance

	Name ▼	Instance	Creation time ▼	Last updated	Status ▼	Lifecycle config	Actions
<input type="radio"/>	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. Nella pagina Jupyter Lab, apri un nuovo **Terminale**.



- Immettere il comando `ssh <nome utente amministratore>@< IP server ONTAP >` per accedere al file system FSx ONTAP . (Il nome utente e l'indirizzo IP vengono recuperati dal passaggio 2) Utilizzare la password utilizzata durante la creazione della **macchina virtuale di archiviazione**.



- Eseguire i comandi nel seguente ordine. Utilizziamo **fsxn-ontap** come nome per il **nome del bucket S3 privato FSx ONTAP ***. Utilizzare ***nome della macchina virtuale di archiviazione** per l'argomento **-vserver**.

```
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
```



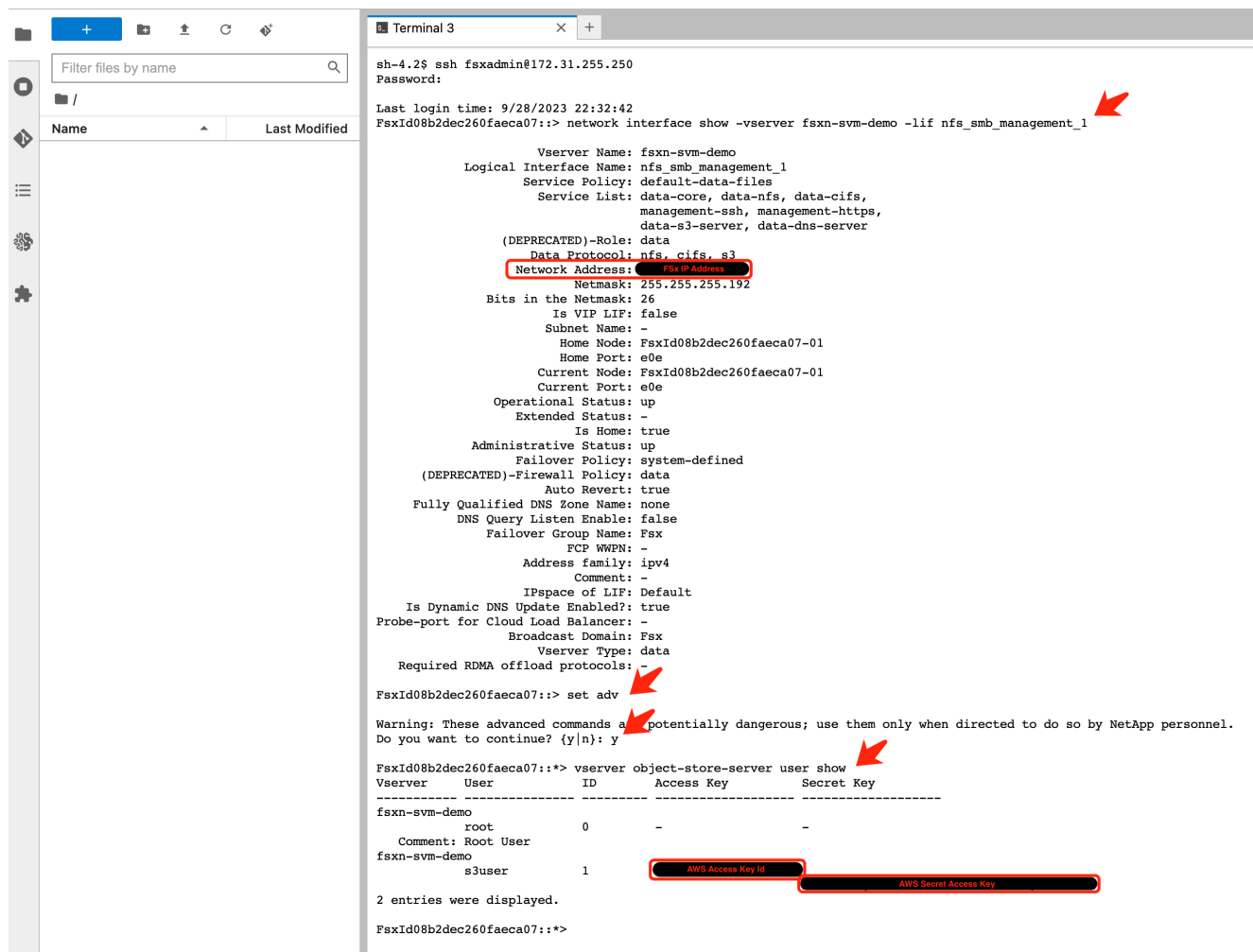
7. Eseguire i comandi seguenti per recuperare l'IP dell'endpoint e le credenziali per FSx ONTAP privato S3.

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

set adv

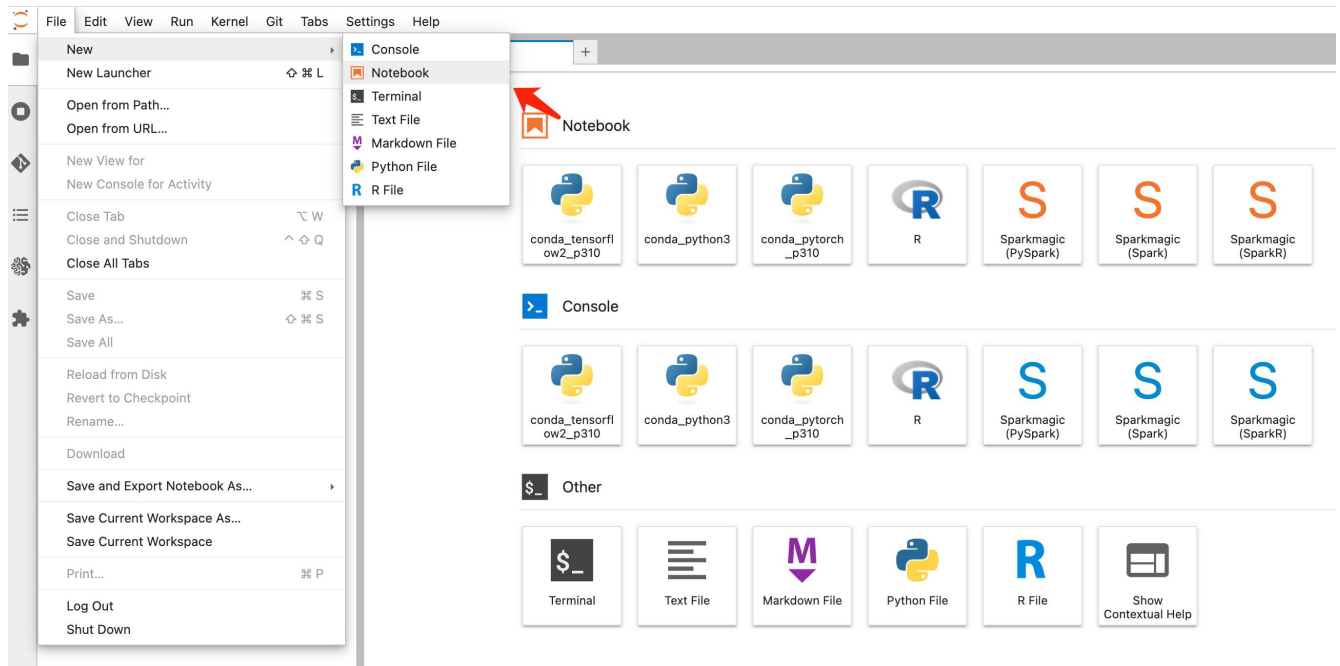
vserver object-store-server user show
```

8. Conservare l'IP e le credenziali dell'endpoint per un utilizzo futuro.



Configurazione del client

1. Nell'istanza di SageMaker Notebook, crea un nuovo notebook Jupyter.



2. Utilizzare il codice seguente come soluzione alternativa per caricare i file nel bucket S3 privato di FSx ONTAP . Per un esempio di codice completo fare riferimento a questo notebook. ["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']

```

Questo conclude l'integrazione tra FSx ONTAP e l'istanza SageMaker.

Lista di controllo utile per il debug

- Assicurarsi che l'istanza di SageMaker Notebook e il file system FSx ONTAP si trovino nella stessa VPC.
- Ricordarsi di eseguire il comando **set dev** su ONTAP per impostare il livello di privilegio su **dev**.

FAQ (aggiornato al 27 settembre 2023)

D: Perché ricevo l'errore **"Si è verificato un errore (NotImplemented) durante la chiamata dell'operazione CreateMultipartUpload: il comando s3 richiesto non è implementato"** quando carico file su FSx ONTAP?

R: In quanto bucket S3 privato, FSx ONTAP supporta il caricamento di file fino a 100 MB. Quando si utilizza il protocollo S3, i file più grandi di 100 MB vengono divisi in blocchi da 100 MB e viene chiamata la funzione 'CreateMultipartUpload'. Tuttavia, l'attuale implementazione di FSx ONTAP private S3 non supporta questa funzione.

D: Perché ricevo l'errore **"Si è verificato un errore (Accesso negato) durante la chiamata delle operazioni PutObject: Accesso negato"** quando carico file su FSx ONTAP?

R: Per accedere al bucket S3 privato FSx ONTAP da un'istanza di SageMaker Notebook, sostituire le credenziali AWS con le credenziali FSx ONTAP. Tuttavia, per concedere l'autorizzazione di scrittura all'istanza è necessaria una soluzione alternativa che prevede il montaggio del bucket e l'esecuzione del comando shell 'chmod' per modificare le autorizzazioni.

D: Come posso integrare il bucket S3 privato FSx ONTAP con altri servizi SageMaker ML?

R: Purtroppo, l'SDK dei servizi SageMaker non fornisce un modo per specificare l'endpoint per il bucket S3 privato. Di conseguenza, FSx ONTAP S3 non è compatibile con i servizi SageMaker quali Sagemaker Data Wrangler, Sagemaker Clarify, Sagemaker Glue, Sagemaker Athena, Sagemaker AutoML e altri.

Parte 2 - Utilizzo di AWS Amazon FSx for NetApp ONTAP (FSx ONTAP) come origine dati per l'addestramento del modello in SageMaker

Questo articolo è un tutorial sull'utilizzo di Amazon FSx for NetApp ONTAP (FSx ONTAP) per l'addestramento di modelli PyTorch in SageMaker, in particolare per un progetto di classificazione della qualità degli pneumatici.

Introduzione

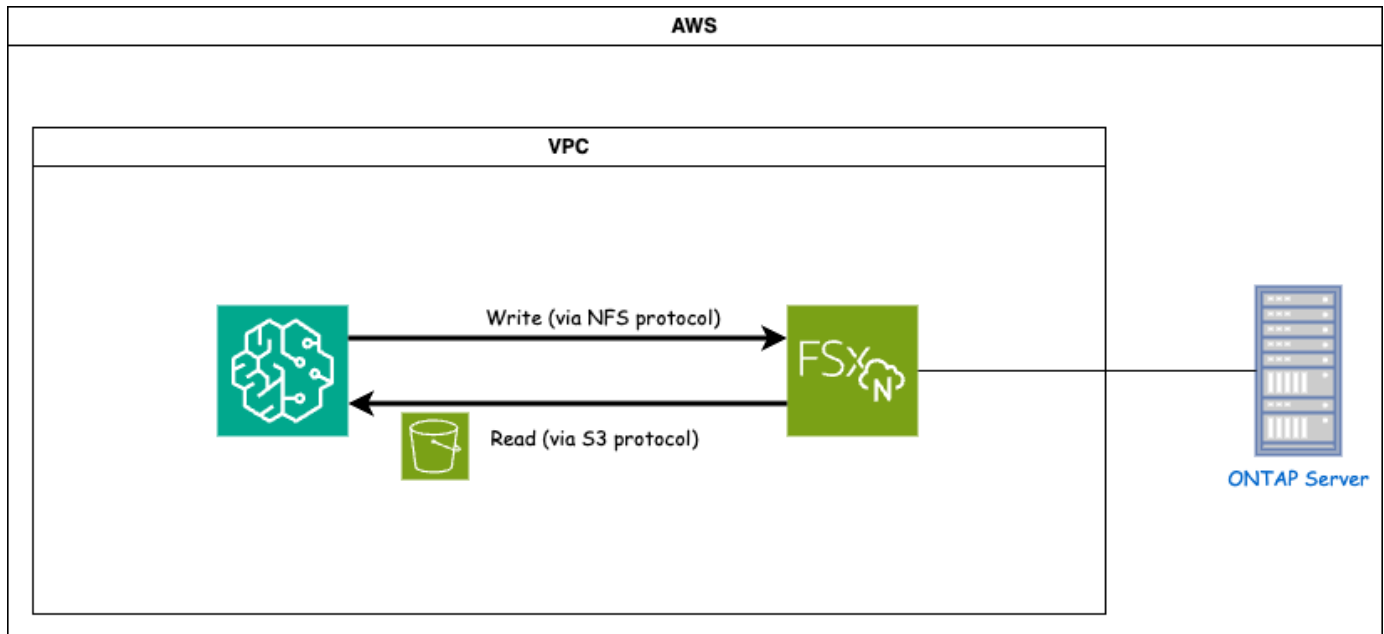
Questo tutorial offre un esempio pratico di un progetto di classificazione della visione artificiale, fornendo un'esperienza pratica nella creazione di modelli di apprendimento automatico che utilizzano FSx ONTAP come origine dati all'interno dell'ambiente SageMaker. Il progetto si concentra sull'utilizzo di PyTorch, un framework di deep learning, per classificare la qualità degli pneumatici in base alle immagini degli stessi. Si concentra sullo sviluppo di modelli di apprendimento automatico utilizzando FSx ONTAP come origine dati in Amazon SageMaker.

Che cos'è FSx ONTAP

Amazon FSx ONTAP è effettivamente una soluzione di storage completamente gestita offerta da AWS. Sfrutta il file system ONTAP di NetApp per fornire uno storage affidabile e ad alte prestazioni. Grazie al supporto di protocolli come NFS, SMB e iSCSI, consente un accesso senza interruzioni da diverse istanze di elaborazione e container. Il servizio è progettato per offrire prestazioni eccezionali, garantendo operazioni sui dati rapide ed efficienti. Offre inoltre elevata disponibilità e durabilità, garantendo che i tuoi dati rimangano accessibili e protetti. Inoltre, la capacità di archiviazione di Amazon FSx ONTAP è scalabile, consentendoti di adattarla facilmente in base alle tue esigenze.

Prerequisito

Ambiente di rete



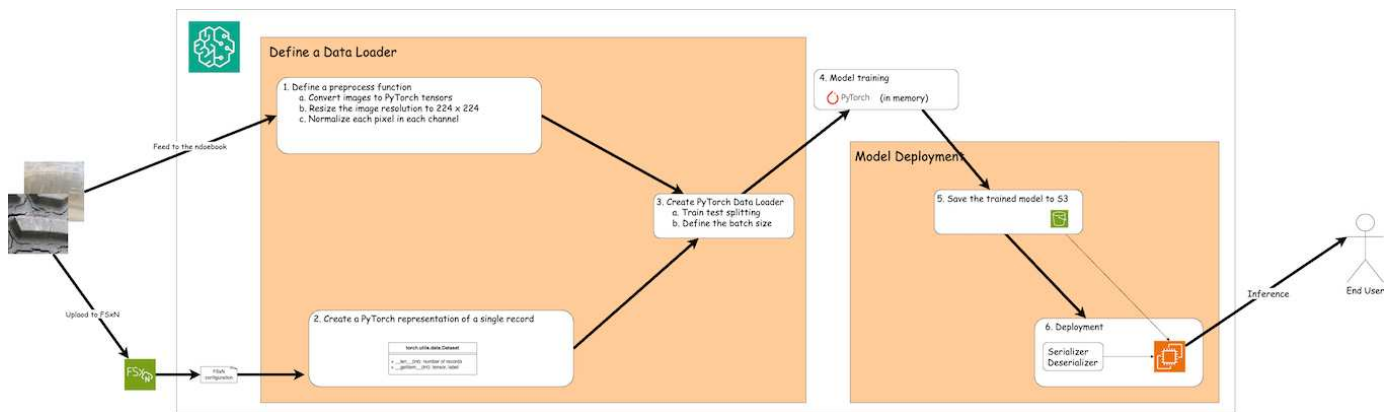
FSx ONTAP (Amazon FSx ONTAP) è un servizio di archiviazione AWS. Include un file system in esecuzione sul sistema NetApp ONTAP e una macchina virtuale (SVM) gestita da AWS che si connette ad esso. Nel diagramma fornito, il server NetApp ONTAP gestito da AWS si trova all'esterno della VPC. L'SVM funge da intermediario tra SageMaker e il sistema NetApp ONTAP, ricevendo le richieste operative da SageMaker e inoltrandole allo storage sottostante. Per accedere a FSx ONTAP, SageMaker deve essere posizionato nella stessa VPC della distribuzione FSx ONTAP. Questa configurazione garantisce la comunicazione e l'accesso ai dati tra SageMaker e FSx ONTAP.

Accesso ai dati

In scenari reali, gli scienziati dei dati in genere utilizzano i dati esistenti archiviati in FSx ONTAP per creare i loro modelli di apprendimento automatico. Tuttavia, a scopo dimostrativo, poiché il file system FSx ONTAP è inizialmente vuoto dopo la creazione, è necessario caricare manualmente i dati di addestramento. Ciò può essere ottenuto montando FSx ONTAP come volume su SageMaker. Una volta montato correttamente il file system, puoi caricare il tuo set di dati nella posizione montata, rendendolo accessibile per l'addestramento dei tuoi modelli nell'ambiente SageMaker. Questo approccio consente di sfruttare la capacità di archiviazione e le funzionalità di FSx ONTAP mentre si lavora con SageMaker per lo sviluppo e la formazione dei modelli.

Il processo di lettura dei dati prevede la configurazione di FSx ONTAP come bucket S3 privato. Per apprendere le istruzioni di configurazione dettagliate, fare riferimento a ["Parte 1 - Integrazione di Amazon FSx for NetApp ONTAP \(FSx ONTAP\) come bucket S3 privato in AWS SageMaker"](#)

Panoramica sull'integrazione



Il flusso di lavoro per l'utilizzo dei dati di training in FSx ONTAP per creare un modello di deep learning in SageMaker può essere riassunto in tre passaggi principali: definizione del caricatore dati, training del modello e distribuzione. Ad alto livello, questi passaggi costituiscono la base di una pipeline MLOps. Tuttavia, ogni fase prevede diversi sotto-fasi dettagliati per un'implementazione completa. Questi sotto-passaggi comprendono varie attività, quali la pre-elaborazione dei dati, la suddivisione del set di dati, la configurazione del modello, l'ottimizzazione degli iperparametri, la valutazione del modello e l'implementazione del modello. Questi passaggi garantiscono un processo completo ed efficace per la creazione e l'implementazione di modelli di deep learning utilizzando i dati di training di FSx ONTAP all'interno dell'ambiente SageMaker.

Integrazione passo dopo passo

Loader dati

Per addestrare una rete di deep learning PyTorch con i dati, viene creato un caricatore di dati per facilitare l'alimentazione dei dati. Il caricatore dati non solo definisce la dimensione del batch, ma determina anche la procedura per la lettura e la preelaborazione di ciascun record all'interno del batch. Configurando il caricatore di dati, possiamo gestire l'elaborazione dei dati in batch, consentendo l'addestramento della rete di deep learning.

Il caricatore dati è composto da 3 parti.

Funzione di pre-elaborazione

```
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]
    )
])
```

Il frammento di codice sopra riportato illustra la definizione delle trasformazioni di pre-elaborazione delle immagini utilizzando il modulo **torchvision.transforms**. In questo tutorial, l'oggetto preprocess viene creato per applicare una serie di trasformazioni. Innanzitutto, la trasformazione **ToTensor()** converte l'immagine in una rappresentazione tensoriale. Successivamente, la trasformazione **Resize 224,224** ridimensiona

l'immagine a una dimensione fissa di 224x224 pixel. Infine, la trasformazione **Normalize()** normalizza i valori del tensore sottraendo la media e dividendo per la deviazione standard lungo ciascun canale. I valori di media e deviazione standard utilizzati per la normalizzazione sono comunemente impiegati nei modelli di reti neurali pre-addestrati. Nel complesso, questo codice prepara i dati dell'immagine per un'ulteriore elaborazione o per l'inserimento in un modello pre-addestrato, convertendoli in un tensore, ridimensionandoli e normalizzando i valori dei pixel.

La classe del set di dati PyTorch

```
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
```

Questa classe fornisce funzionalità per ottenere il numero totale di record nel set di dati e definisce il metodo per leggere i dati per ciascun record. All'interno della funzione *getitem*, il codice utilizza l'oggetto bucket S3 boto3 per recuperare i dati binari da FSx ONTAP. Lo stile del codice per l'accesso ai dati da FSx ONTAP è simile a quello per la lettura dei dati da Amazon S3. La spiegazione successiva approfondisce il processo di creazione dell'oggetto S3 privato **bucket**.

FSx ONTAP come repository S3 privato

```
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 FsxN 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 ---
```

Per leggere i dati da FSx ONTAP in SageMaker, viene creato un gestore che punta allo storage FSx ONTAP utilizzando il protocollo S3. Ciò consente di trattare FSx ONTAP come un bucket S3 privato. La configurazione del gestore include la specifica dell'indirizzo IP dell'SVM FSx ONTAP, del nome del bucket e delle credenziali necessarie. Per una spiegazione completa su come ottenere questi elementi di configurazione, fare riferimento al documento all'indirizzo ["Parte 1 - Integrazione di Amazon FSx for NetApp ONTAP \(FSx ONTAP\) come bucket S3 privato in AWS SageMaker"](#).

Nell'esempio menzionato sopra, l'oggetto bucket viene utilizzato per istanziare l'oggetto dataset PyTorch. L'oggetto dataset verrà spiegato più dettagliatamente nella sezione successiva.

II Loader dati PyTorch

```
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)
```

Nell'esempio fornito, viene specificata una dimensione batch pari a 64, a indicare che ogni batch conterrà 64 record. Combinando la classe PyTorch **Dataset**, la funzione di pre-elaborazione e la dimensione del batch di addestramento, otteniamo il caricatore di dati per l'addestramento. Questo caricatore di dati semplifica il processo di iterazione del set di dati in batch durante la fase di addestramento.

Formazione del modello

```
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')

```

Questo codice implementa un processo di addestramento PyTorch standard. Definisce un modello di rete neurale denominato **TyreQualityClassifier** che utilizza livelli convoluzionali e un livello lineare per classificare la qualità degli pneumatici. Il ciclo di addestramento esegue iterazioni su batch di dati, calcola la perdita e aggiorna i parametri del modello utilizzando la backpropagation e l'ottimizzazione. Inoltre, stampa l'ora corrente, l'epoca, il lotto e la perdita a scopo di monitoraggio.

Distribuzione del modello

Distribuzione

```

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())

```

Il codice salva il modello PyTorch su **Amazon S3** perché SageMaker richiede che il modello venga archiviato in S3 per la distribuzione. Caricando il modello su **Amazon S3**, questo diventa accessibile a SageMaker, consentendo la distribuzione e l'inferenza sul modello distribuito.

```

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
)

```

Questo codice facilita l'implementazione di un modello PyTorch su SageMaker. Definisce un serializzatore personalizzato, **TyreQualitySerializer**, che preelabora e serializza i dati di input come un tensore PyTorch. La classe **TyreQualityPredictor** è un predittore personalizzato che utilizza il serializzatore definito e un **JSONDeserializer**. Il codice crea anche un oggetto **PyTorchModel** per specificare la posizione S3 del modello, il ruolo IAM, la versione del framework e il punto di ingresso per l'inferenza. Il codice genera un

timestamp e costruisce un nome di endpoint basato sul modello e sul timestamp. Infine, il modello viene distribuito utilizzando il metodo `deploy`, specificando il conteggio delle istanze, il tipo di istanza e il nome dell'endpoint generato. Ciò consente di distribuire il modello PyTorch e di renderlo accessibile per l'inferenza su SageMaker.

Inferenza

```
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)
```

Questo è un esempio di utilizzo dell'endpoint distribuito per eseguire l'inferenza.

Parte 3 - Creazione di una pipeline MLOps semplificata (CI/CT/CD)

Questo articolo fornisce una guida alla creazione di una pipeline MLOps con i servizi AWS, concentrandosi sulla riqualificazione automatizzata dei modelli, sulla distribuzione e sull'ottimizzazione dei costi.

Introduzione

In questo tutorial imparerai come sfruttare vari servizi AWS per costruire una semplice pipeline MLOps che comprende integrazione continua (CI), formazione continua (CT) e distribuzione continua (CD). A differenza delle pipeline DevOps tradizionali, MLOps richiede considerazioni aggiuntive per completare il ciclo operativo. Seguendo questo tutorial, imparerai come integrare CT nel ciclo MLOps, consentendo un addestramento continuo dei tuoi modelli e una distribuzione senza interruzioni per l'inferenza. Il tutorial ti guiderà attraverso il processo di utilizzo dei servizi AWS per stabilire questa pipeline MLOps end-to-end.

Manifesto

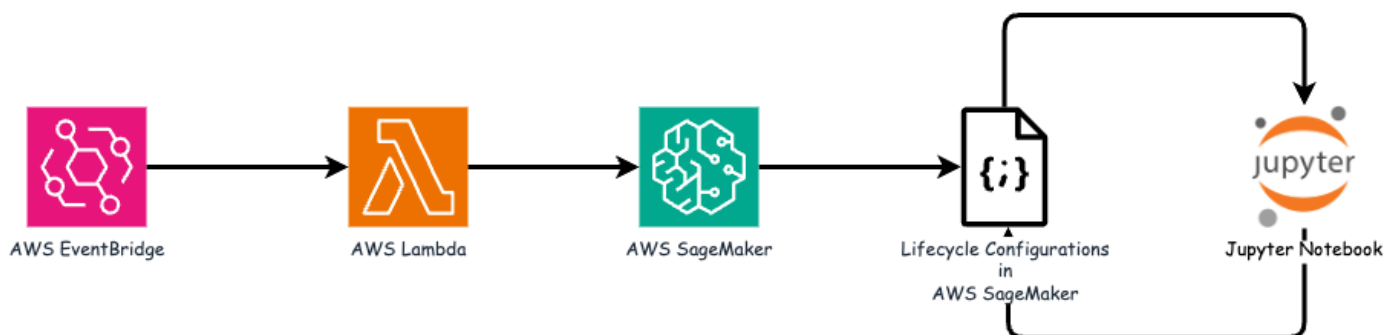
Funzionalità	Nome	Commento
Archiviazione dei dati	AWS FSx ONTAP	Fare riferimento a "Parte 1 - Integrazione di Amazon FSx for NetApp ONTAP (FSx ONTAP) come bucket S3 privato in AWS SageMaker" .
IDE di scienza dei dati	AWS SageMaker	Questo tutorial si basa sul notebook Jupyter presentato in "Parte 2 - Utilizzo di Amazon FSx for NetApp ONTAP (FSx ONTAP) come origine dati per l'addestramento del modello in SageMaker" .

Funzionalità	Nome	Commento
Funzione per attivare la pipeline MLOps	Funzione AWS Lambda	-
Trigger del lavoro cron	AWS EventBridge	-
Framework di apprendimento profondo	PyTorch	-
SDK Python di AWS	boto3	-
Linguaggio di programmazione	Pitone	v3.10

Prerequisito

- Un file system FSx ONTAP preconfigurato. Questo tutorial utilizza i dati memorizzati in FSx ONTAP per il processo di formazione.
- Un'istanza di **SageMaker Notebook** configurata per condividere la stessa VPC del file system FSx ONTAP menzionato sopra.
- Prima di attivare la **funzione AWS Lambda**, assicurarsi che l'istanza **SageMaker Notebook** sia nello stato **arrestato**.
- Il tipo di istanza **ml.g4dn.xlarge** è necessario per sfruttare l'accelerazione GPU necessaria per i calcoli delle reti neurali profonde.

Architettura



Questa pipeline MLOps è un'implementazione pratica che utilizza un cron job per attivare una funzione serverless, che a sua volta esegue un servizio AWS registrato con una funzione di callback del ciclo di vita. **AWS EventBridge** funge da cron job. Periodicamente richiama una **funzione AWS Lambda** responsabile della riqualificazione e della ridistribuzione del modello. Questo processo prevede l'avvio dell'istanza **AWS SageMaker Notebook** per eseguire le attività necessarie.

Configurazione passo passo

Configurazioni del ciclo di vita

Per configurare la funzione di callback del ciclo di vita per l'istanza AWS SageMaker Notebook, è necessario utilizzare **Configurazioni del ciclo di vita**. Questo servizio consente di definire le azioni necessarie da eseguire durante l'avvio dell'istanza del notebook. Nello specifico, è possibile implementare uno script shell all'interno delle **configurazioni del ciclo di vita** per arrestare automaticamente l'istanza del notebook una volta completati i processi di formazione e distribuzione. Questa è una configurazione obbligatoria poiché il

costo è uno degli aspetti più importanti da considerare in MLOps.

È importante notare che la configurazione per le **configurazioni del ciclo di vita** deve essere impostata in anticipo. Pertanto, si consiglia di dare priorità alla configurazione di questo aspetto prima di procedere con l'altra configurazione della pipeline MLOps.

1. Per impostare le configurazioni del ciclo di vita, apri il pannello **Sagemaker** e vai a **Configurazioni del ciclo di vita** nella sezione **Configurazioni amministrative**.

aws

Services

Q 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

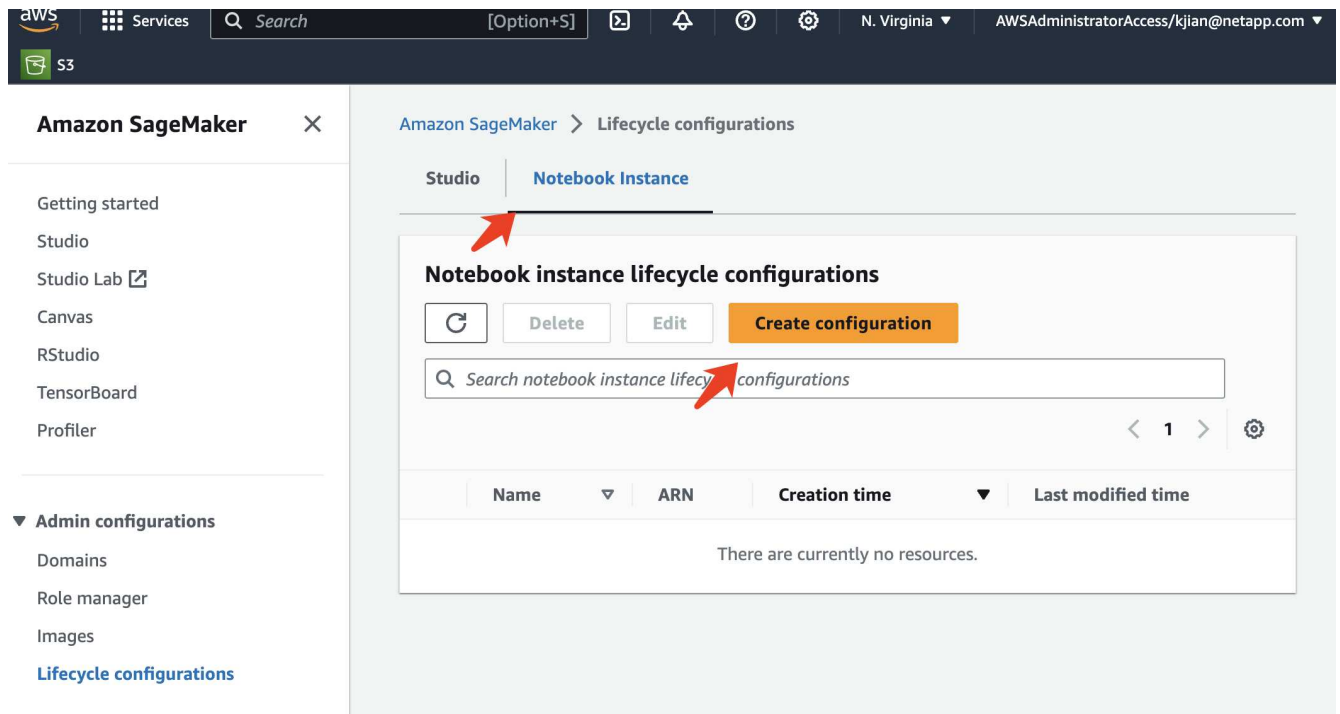
Domains (4)

Info

Q 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. Selezionare la scheda **Istanza notebook** e fare clic sul pulsante **Crea configurazione**



3. Incolla il codice sottostante nell'area di immissione.

```
#!/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. Questo script esegue Jupyter Notebook, che gestisce il riaddestramento e la ridistribuzione del modello per l'inferenza. Una volta completata l'esecuzione, il notebook si spegnerà automaticamente entro 5 minuti. Per saperne di più sulla definizione del problema e sull'implementazione del codice, fare riferimento a ["Parte 2 - Utilizzo di Amazon FSx for NetApp ONTAP \(FSx ONTAP\) come origine dati per l'addestramento del modello in SageMaker"](#).

aws Services Search [Option+S]

S3

Amazon SageMaker > Lifecycle configurations > Create lifecycle configuration

Create lifecycle configuration

Configuration setting

Name

fsxn-demo-lifecycle-callback

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

Scripts

Start notebook Create notebook

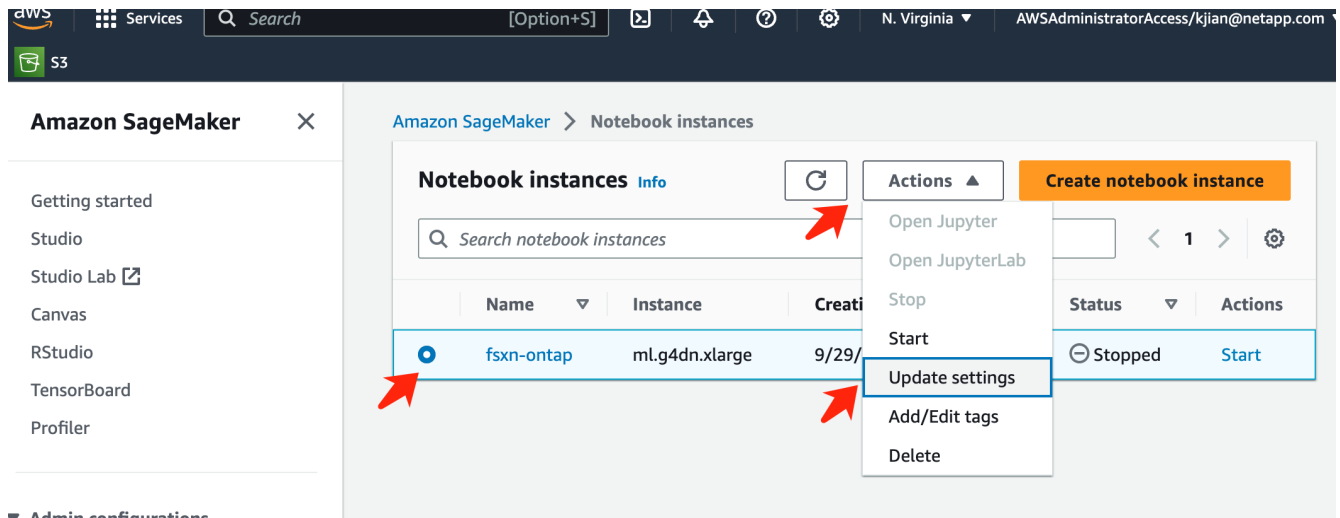
This script will be run each time an associated notebook instance is started, including during initial creation. If the associated notebook instance is already started, it will be run the next time it is stopped and started. [a curated list of sample scripts](#)

```
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 torch_p310
9 nohup jupyter nbconvert "$NOTEBOOK_FILE" --ExecutePreprocessor.kernel_name=python --execute --to nbconvert_pid=$!
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
```

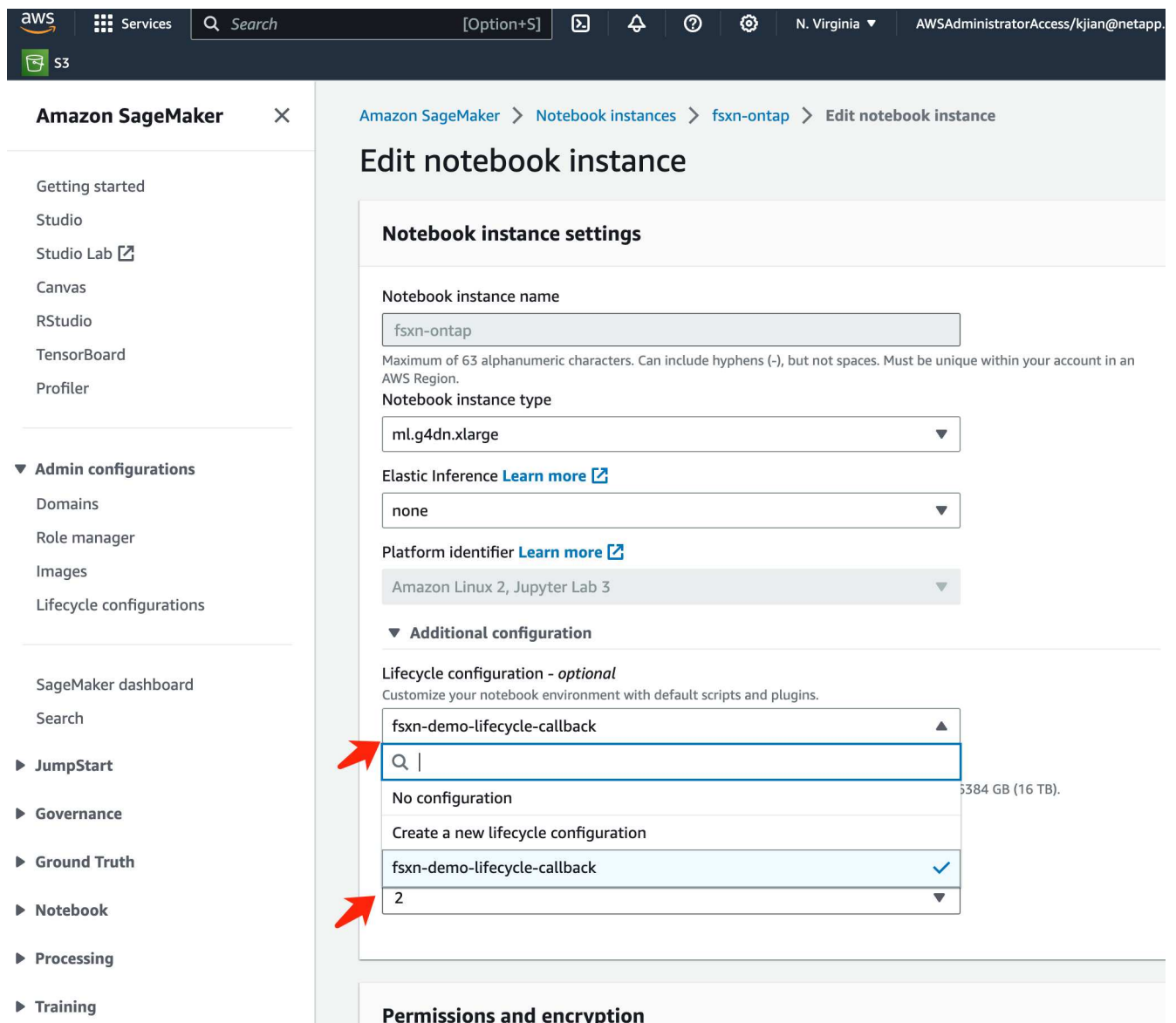
Cancel Create configuration

CloudShell Feedback

5. Dopo la creazione, vai alle istanze del Notebook, seleziona l'istanza di destinazione e fai clic su **Aggiorna impostazioni** nel menu a discesa Azioni.



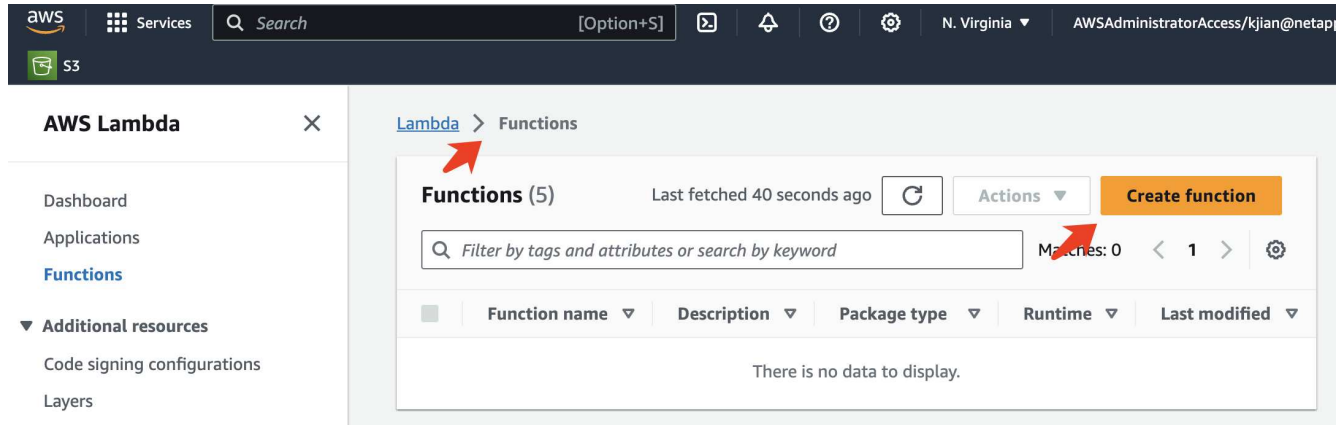
6. Selezionare la **Configurazione del ciclo di vita** creata e fare clic su **Aggiorna istanza del notebook**.



Funzione serverless AWS Lambda

Come accennato in precedenza, la **funzione AWS Lambda** è responsabile dell'avvio dell'istanza **AWS SageMaker Notebook**.

1. Per creare una **funzione AWS Lambda**, vai al pannello corrispondente, passa alla scheda **Funzioni** e fai clic su **Crea funzione**.



2. Si prega di compilare tutte le voci richieste nella pagina e di ricordarsi di impostare il Runtime su **Python 3.10**.

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. Verificare che il ruolo designato disponga dell'autorizzazione richiesta **AmazonSageMakerFullAccess** e fare clic sul pulsante **Crea funzione**.

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. Selezionare la funzione Lambda creata. Nella scheda codice, copia e incolla il seguente codice nell'area di testo. Questo codice avvia l'istanza del notebook denominata **fsxn-ontap**.

```
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. Fare clic sul pulsante **Distribuisci** per applicare questa modifica al codice.

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 'N. Virgin'. The main content area is divided into two sections. The top section shows the function name 'demo-mlops' and a 'Layers' section with '(0)' layers. Below this are two buttons: '+ Add trigger' and '+ Add destination'. To the right, there is a summary of the function: 'Last modified 1 minute ago', 'Function ARN: arn:aws:lambda:us-east-1:232233133319:function:fsxn-demo-mlops', and 'Function URL: Info'. The bottom section has tabs for 'Code', 'Test', 'Monitor', 'Configuration', 'Aliases', and 'Versions'. The 'Code source' tab is selected, showing a code editor with a Python script. The script imports 'boto3' and 'logging', and defines a 'lambda_handler' function that uses 'boto3.client' to interact with 'sagemaker'. A red arrow points to the 'Test' button in the top right of the code editor area. The code editor also has a 'Deploy' button and a 'Changes not deployed' status indicator.

```
1 import boto3
2 import logging
3
4 def lambda_handler(event, context):
5     client = boto3.client('sagemaker')
6     logging.info('Invoking SageMaker')
7     client.start_notebook_instance(NotebookInstanceName='fsxn-ontap')
8     return {
9         'statusCode': 200,
10        'body': f'Starting notebook instance: {notebook_instance_name}'
11    }
12
```

6. Per specificare come attivare questa funzione AWS Lambda, fare clic sul pulsante Aggiungi trigger.

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. Selezionare EventBridge dal menu a discesa, quindi fare clic sul pulsante di opzione denominato Crea una nuova regola. Nel campo espressione pianificazione, immettere `rate (1 day)` e fai clic sul pulsante Aggiungi per creare e applicare questa nuova regola cron job alla funzione AWS Lambda.

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

Dopo aver completato la configurazione in due fasi, ogni giorno la **funzione AWS Lambda** avvierà **SageMaker Notebook**, eseguirà il riaddestramento del modello utilizzando i dati dal repository **FSx ONTAP**, ridistribuirà il modello aggiornato nell'ambiente di produzione e arresterà automaticamente l'**istanza di SageMaker Notebook** per ottimizzare i costi. Ciò garantisce che il modello rimanga aggiornato.

Si conclude qui il tutorial sullo sviluppo di una pipeline MLOps.

Informazioni sul copyright

Copyright © 2026 NetApp, Inc. Tutti i diritti riservati. Stampato negli Stati Uniti d'America. Nessuna porzione di questo documento soggetta a copyright può essere riprodotta in qualsiasi formato o mezzo (grafico, elettronico o meccanico, inclusi fotocopie, registrazione, nastri o storage in un sistema elettronico) senza previo consenso scritto da parte del detentore del copyright.

Il software derivato dal materiale sottoposto a copyright di NetApp è soggetto alla seguente licenza e dichiarazione di non responsabilità:

IL PRESENTE SOFTWARE VIENE FORNITO DA NETAPP "COSÌ COM'È" E SENZA QUALSIVOGLIA TIPO DI GARANZIA IMPLICITA O ESPRESSA FRA CUI, A TITOLO ESEMPLIFICATIVO E NON ESAUSTIVO, GARANZIE IMPLICITE DI COMMERCIALIZZABILITÀ E IDONEITÀ PER UNO SCOPO SPECIFICO, CHE VENGONO DECLINATE DAL PRESENTE DOCUMENTO. NETAPP NON VERRÀ CONSIDERATA RESPONSABILE IN ALCUN CASO PER QUALSIVOGLIA DANNO DIRETTO, INDIRETTO, ACCIDENTALE, SPECIALE, ESEMPLARE E CONSEGUENZIALE (COMPRESI, A TITOLO ESEMPLIFICATIVO E NON ESAUSTIVO, PROCUREMENT O SOSTITUZIONE DI MERCI O SERVIZI, IMPOSSIBILITÀ DI UTILIZZO O PERDITA DI DATI O PROFITTI OPPURE INTERRUZIONE DELL'ATTIVITÀ AZIENDALE) CAUSATO IN QUALSIVOGLIA MODO O IN RELAZIONE A QUALUNQUE TEORIA DI RESPONSABILITÀ, SIA ESSA CONTRATTUALE, RIGOROSA O DOVUTA A INSOLVENZA (COMPRESA LA NEGLIGENZA O ALTRO) INSORTA IN QUALSIASI MODO ATTRAVERSO L'UTILIZZO DEL PRESENTE SOFTWARE ANCHE IN PRESENZA DI UN PREAVVISO CIRCA L'EVENTUALITÀ DI QUESTO TIPO DI DANNI.

NetApp si riserva il diritto di modificare in qualsiasi momento qualunque prodotto descritto nel presente documento senza fornire alcun preavviso. NetApp non si assume alcuna responsabilità circa l'utilizzo dei prodotti o materiali descritti nel presente documento, con l'eccezione di quanto concordato espressamente e per iscritto da NetApp. L'utilizzo o l'acquisto del presente prodotto non comporta il rilascio di una licenza nell'ambito di un qualche diritto di brevetto, marchio commerciale o altro diritto di proprietà intellettuale di NetApp.

Il prodotto descritto in questa guida può essere protetto da uno o più brevetti degli Stati Uniti, esteri o in attesa di approvazione.

LEGENDA PER I DIRITTI SOTTOPOSTI A LIMITAZIONE: l'utilizzo, la duplicazione o la divulgazione da parte degli enti governativi sono soggetti alle limitazioni indicate nel sottoparagrafo (b)(3) della clausola Rights in Technical Data and Computer Software del DFARS 252.227-7013 (FEB 2014) e FAR 52.227-19 (DIC 2007).

I dati contenuti nel presente documento riguardano un articolo commerciale (secondo la definizione data in FAR 2.101) e sono di proprietà di NetApp, Inc. Tutti i dati tecnici e il software NetApp forniti secondo i termini del presente Contratto sono articoli aventi natura commerciale, sviluppati con finanziamenti esclusivamente privati. Il governo statunitense ha una licenza irrevocabile limitata, non esclusiva, non trasferibile, non cedibile, mondiale, per l'utilizzo dei Dati esclusivamente in connessione con e a supporto di un contratto governativo statunitense in base al quale i Dati sono distribuiti. Con la sola esclusione di quanto indicato nel presente documento, i Dati non possono essere utilizzati, divulgati, riprodotti, modificati, visualizzati o mostrati senza la previa approvazione scritta di NetApp, Inc. I diritti di licenza del governo degli Stati Uniti per il Dipartimento della Difesa sono limitati ai diritti identificati nella clausola DFARS 252.227-7015(b) (FEB 2014).

Informazioni sul marchio commerciale

NETAPP, il logo NETAPP e i marchi elencati alla pagina <http://www.netapp.com/TM> sono marchi di NetApp, Inc. Gli altri nomi di aziende e prodotti potrebbero essere marchi dei rispettivi proprietari.