



Cloud Data Management with NetApp File-Object Duality and AWS SageMaker

NetApp Solutions

NetApp
May 03, 2023

Table of Contents

- Cloud Data Management with NetApp File-Object Duality and AWS SageMaker 1
 - TR-4967: Cloud Data Management with NetApp File-Object Duality and AWS SageMaker 1
 - Solution technology 1
 - Data duality for data scientists and other applications 2
 - Conclusion 29

Cloud Data Management with NetApp File-Object Duality and AWS SageMaker

TR-4967: Cloud Data Management with NetApp File-Object Duality and AWS SageMaker

Karthikeyan Nagalingam, NetApp

Data scientists and engineers often need to access data stored in the NFS format, but accessing this data directly from the S3 protocol in AWS SageMaker can be challenging because AWS only supports S3 bucket access. However, NetApp ONTAP provides a solution by enabling dual-protocol access for NFS and S3. With this solution, data scientists and engineers can access NFS data from AWS SageMaker notebooks via S3 buckets from NetApp Cloud Volumes ONTAP. This approach enables easy access and sharing of the same data from both NFS and S3 without the need for additional software.

[Next: Solution technology.](#)

Solution technology

[Previous: Solution overview.](#)

This solution utilizes the following technologies:

- **AWS SageMaker Notebook.** Offers machine learning capabilities to developers and data scientists to create, train, and deploy high-quality ML models efficiently.
- **NetApp BlueXP.** Enables the discovery, deployment, and operation of storage on premises as well as on AWS, Azure, and Google Cloud. It provides data protection against data loss, cyber threats, and unplanned outages and optimizes data storage and infrastructure.
- **NetApp Cloud Volumes ONTAP.** Provides enterprise-grade storage volumes with NFS, SMB/CIFS, iSCSI, and S3 protocols on AWS, Azure, and Google Cloud, giving users greater flexibility in accessing and managing their data in the cloud.

NetApp Cloud Volumes ONTAP created from BlueXP to store ML data.

The following figure shows the technical components of the solution.



Use case summary

A potential use case for dual protocol access of NFS and S3 is in the fields of machine learning and data science. For example, a team of data scientists might be working on a machine learning project using AWS SageMaker, which requires access to data stored in the NFS format. However, the data might also need to be accessed and shared via S3 buckets to collaborate with other team members or to integrate with other applications that use S3.

By utilizing NetApp Cloud Volumes ONTAP, the team can store their data in a single location and have it accessible with both NFS and S3 protocols. The data scientists can access the data in NFS format directly from AWS SageMaker, while other team members or applications can access the same data via S3 buckets.

This approach enables the data to be accessed and shared easily and efficiently without the need for additional software or data migration between different storage solutions. It also allows for a more streamlined workflow and collaboration among team members, resulting in faster and more effective development of machine learning models.

[Next: Data duality for data scientists and other applications.](#)

Data duality for data scientists and other applications

[Previous: Solution technology.](#)

Data is available in NFS and accessed from S3 from AWS SageMaker.

Technology requirements

You need NetApp BlueXP, NetApp Cloud Volumes ONTAP, and AWS SageMaker Notebooks for the data-duality use case.

Software requirements

The following table lists the software components that are required to implement the use case.

Software	Quantity
BlueXP	1

Software	Quantity
NetApp Cloud Volumes ONTAP	1
AWS SageMaker Notebook	1

Deployment procedures

Deploying the data-duality solution involves the following tasks:

- BlueXP Connector
- NetApp Cloud Volumes ONTAP
- Data for machine learning
- AWS SageMaker
- Validated machine learning from Jupyter Notebooks

BlueXP connector

In this validation, we used AWS. It's also applicable for Azure and Google Cloud. To create a BlueXP Connector in AWS, complete the following steps:

1. We used the credentials based on the mcarl-marketplace-subscription in BlueXP.
2. Choose the region suitable for your environment (for example, us-east-1 [N. Virginia]), and select the authentication method (for example, Assume Role or AWS keys). In this validation, we use AWS keys.
3. Provide the name of the connector and create a role.
4. Provide the network details such as the VPC, subnet, or keypair, depending on whether you need a public IP or not.
5. Provide the details for the security group, such as HTTP, HTTPS, or SSH access from the source type, such as anywhere and IP range information.
6. Review and create the BlueXP Connector.
7. Verify that the BlueXP EC2 instance state is running in the AWS console, and check the IP address from the **Networking** tab.
8. Log into the connector user interface from the BlueXP portal, or you can use the IP address for access from the browser.

NetApp Cloud Volumes ONTAP

To create a Cloud Volumes ONTAP instance in BlueXP, complete the following steps:

1. Create a new working environment, select the cloud provider, and select the type of Cloud Volumes ONTAP instance, (such as single-CVO, HA, or Amazon FSxN for ONTAP).
2. Provide details such as the Cloud Volumes ONTAP cluster name and credentials. In this validation, we created a Cloud Volumes ONTAP instance called `svm_sagemaker_cvo_sn1`.
3. Select the services needed for Cloud Volumes ONTAP. In this validation, we choose to only monitor, so we disabled **Data Sense & Compliance** and **Backup to Cloud Services**.
4. In the **Location & Connectivity** section, select the AWS region, VPC, subnet, security group, SSH authentication method, and either a password or a key pair.

5. Choose the charging method. We used **Professional** for this validation.
6. You can choose a preconfigured package, such as **POC and Small Workloads**, **Database and Application Data Production Workloads**, **Cost Effective DR**, or **Highest Performance Production Workloads**. In this validation, we choose **Poc and Small Workloads**.
7. Create a volume with a specific size, allowed protocols, and export options. In this validation, we created a volume called `vol1`.
8. Choose a profile disk type and tiering policy. In this validation, we disabled **Storage Efficiency** and **General- Purpose SSD – Dynamic Performance**.
9. Finally, review and create the Cloud Volumes ONTAP instance. Then wait for 15-20 minutes for BlueXP to create the Cloud Volumes ONTAP working environment.
10. Configure the following parameters to enable the Duality protocol. The Duality protocol (NFS/S3) is supported from ONTAP 9. 12.1 and later.
 - a. In this validation, we created an SVM called `svm_sagemaker_cvo_sn1` and volume `vol1`.
 - b. Verify that the SVM has the protocol support for NFS and S3. If not, modify the SVM to support them.

```

sagemaker_cvo_sn1::> vserver show -vserver svm_sagemaker_cvo_sn1
                                Vserver: svm_sagemaker_cvo_sn1
                                Vserver Type: data
                                Vserver Subtype: default
                                Vserver UUID: 911065dd-a8bc-11ed-bc24-
e1c0f00ad86b
                                Root Volume:
svm_sagemaker_cvo_sn1_root
                                Aggregate: aggr1
                                NIS Domain: -
                                Root Volume Security Style: unix
                                LDAP Client: -
                                Default Volume Language Code: C.UTF-8
                                Snapshot Policy: default
                                Data Services: data-cifs, data-
flexcache,
                                data-iscsi, data-nfs,
                                data-nvme-tcp
                                Comment:
                                Quota Policy: default
                                List of Aggregates Assigned: aggr1
                                Limit on Maximum Number of Volumes allowed: unlimited
                                Vserver Admin State: running
                                Vserver Operational State: running
                                Vserver Operational State Stopped Reason: -
                                Allowed Protocols: nfs, cifs, fcp, iscsi,
ndmp, s3
                                Disallowed Protocols: nvme
                                Is Vserver with Infinite Volume: false
                                QoS Policy Group: -
                                Caching Policy Name: -
                                Config Lock: false
                                IPspace Name: Default
                                Foreground Process: -
                                Logical Space Reporting: true
                                Logical Space Enforcement: false
                                Default Anti_ransomware State of the Vserver's Volumes: disabled
                                Enable Analytics on New Volumes: false
                                Enable Activity Tracking on New Volumes: false

sagemaker_cvo_sn1::>

```

11. Create and install a CA certificate if required.

12. Create a service data policy.

```
sagemaker_cvo_sn1::*> network interface service-policy create -vserver
svm_sagemaker_cvo_sn1 -policy sagemaker_s3_nfs_policy -services data-
core,data-s3-server,data-nfs,data-flexcache
sagemaker_cvo_sn1::*> network interface create -vserver
svm_sagemaker_cvo_sn1 -lif svm_sagemaker_cvo_sn1_s3_lif -service-policy
sagemaker_s3_nfs_policy -home-node sagemaker_cvo_sn1-01 -address
172.30.10.41 -netmask 255.255.255.192
```

Warning: The configured failover-group has no valid failover targets for the LIF's failover-policy. To view the failover targets for a LIF, use the "network interface show -failover" command.

```
sagemaker_cvo_sn1::*>
```

```
sagemaker_cvo_sn1::*> network interface show
```

Logical Vserver Home	Status Interface	Network Admin/Oper	Current Address/Mask	Current Is Node	Is Port

sagemaker_cvo_sn1	cluster-mgmt	up/up	172.30.10.40/26	sagemaker_cvo_sn1-	
01					e0a
true					
	intercluster	up/up	172.30.10.48/26	sagemaker_cvo_sn1-	
01					e0a
true					
	sagemaker_cvo_sn1-01_mgmt1	up/up	172.30.10.58/26	sagemaker_cvo_sn1-	
01					e0a
true					
svm_sagemaker_cvo_sn1	svm_sagemaker_cvo_sn1_data_lif	up/up	172.30.10.23/26	sagemaker_cvo_sn1-	
01					e0a
true					
	svm_sagemaker_cvo_sn1_mgmt_lif	up/up	172.30.10.32/26	sagemaker_cvo_sn1-	
01					e0a
true					
	svm_sagemaker_cvo_sn1_s3_lif	up/up	172.30.10.41/26	sagemaker_cvo_sn1-	

01

e0a

true

6 entries were displayed.

sagemaker_cvo_sn1:*>

```
sagemaker_cvo_sn1:*> vsserver object-store-server create -vsserver  
svm_sagemaker_cvo_sn1 -is-http-enabled true -object-store-server  
svm_sagemaker_cvo_s3_sn1 -is-https-enabled false  
sagemaker_cvo_sn1:*> vsserver object-store-server show
```

Vserver: svm_sagemaker_cvo_sn1

Object Store Server Name: svm_sagemaker_cvo_s3_sn1

Administrative State: up

HTTP Enabled: true

Listener Port For HTTP: 80

HTTPS Enabled: false

Secure Listener Port For HTTPS: 443

Certificate for HTTPS Connections: -

Default UNIX User: pcuser

Default Windows User: -

Comment:

sagemaker_cvo_sn1:*>

13. Check the aggregate details.

```
sagemaker_cvo_sn1::*> aggr show
```

Aggregate Status	Size	Available	Used%	State	#Vols	Nodes	RAID
-----	-----	-----	-----	-----	-----	-----	-----
aggr0_sagemaker_cvo_sn1_01	124.0GB	50.88GB	59%	online	1	sagemaker_cvo_	
raid0,						sn1-01	
normal							
aggr1	907.1GB	904.9GB	0%	online	2	sagemaker_cvo_	
raid0,						sn1-01	
normal							

2 entries were displayed.

```
sagemaker_cvo_sn1::*>
```

14. Create a user and group.

```
sagemaker_cvo_sn1:*> vservers object-store-server user create -vservers
svm_sagemaker_cvo_sn1 -user s3user

sagemaker_cvo_sn1:*> vservers object-store-server user show
Vserver      User      ID      Access Key      Secret Key
-----
svm_sagemaker_cvo_sn1
      root      0      -      -
      Comment: Root User
svm_sagemaker_cvo_sn1
      s3user      1      0ZNAX21JW5Q8AP80CQ2E
PpLs4gA9K0_2gPhuykkp014gBjcC9Rbi3QDX_6rr
2 entries were displayed.

sagemaker_cvo_sn1:*>

sagemaker_cvo_sn1:*> vservers object-store-server group create -name
s3group -users s3user -comment ""

sagemaker_cvo_sn1:*>
sagemaker_cvo_sn1:*> vservers object-store-server group delete -gid 1
-vservers svm_sagemaker_cvo_sn1

sagemaker_cvo_sn1:*> vservers object-store-server group create -name
s3group -users s3user -comment "" -policies FullAccess

sagemaker_cvo_sn1:*>
```

15. Create a bucket on the NFS volume.

```
sagemaker_cvo_sn1::~*> vsriver object-store-server bucket create -bucket
ontapbucket1 -type nas -comment "" -vsriver svm_sagemaker_cvo_sn1 -nas
-path /vol1
sagemaker_cvo_sn1::~*> vsriver object-store-server bucket show
Vserver      Bucket      Type      Volume      Size
Encryption Role      NAS Path
-----
svm_sagemaker_cvo_sn1
ontapbucket1 nas      vol1      -      false
-      /vol1
sagemaker_cvo_sn1::~*>
```

AWS SageMaker

To create an AWS Notebook from AWS SageMaker, complete the following steps:

1. Make sure the user who is creating Notebook instance has an AmazonSageMakerFullAccess IAM policy or is part of an existing group that has AmazonSageMakerFullAccess rights. In this validation, the user is part of an existing group.
2. Provide the following information:
 - Notebook instance name.
 - Instance type.
 - Platform identifier.
 - Select the IAM role that has AmazonSageMakerFullAccess rights.
 - Root access – enable.
 - Encryption key - Select no custom encryption.
 - Keep the remaining default options.
3. In this validation, the SageMaker instance details are as follows:

Amazon SageMaker > Notebook instances > nkarthiksagemaker

nkarthiksagemaker

Delete

Stop

Open Jupyter

Open JupyterLab

Notebook instance settings

Edit

Name nkarthiksagemaker	Status ✔ InService	Notebook instance type ml.t2.medium	Platform identifier Amazon Linux 2, Jupyter Lab 3 (notebook-al2-v2)
ARN arn:aws:sagemaker:us-east-1:210811600188:notebook-instance/nkarthiksagemaker	Creation time Feb 16, 2023 18:55 UTC	Elastic Inference -	Minimum IMDS Version 2
Lifecycle configuration -	Last updated Mar 22, 2023 20:59 UTC	Volume Size 5GB EBS	

Permissions and encryption

IAM role ARN

[arn:aws:iam::210811600188:role/SageMakerFullRole](#)

Root access

Enabled

Encryption key

Network

Subnet(s)

[subnet-00f94558](#)

Security Group(s)

[sg-07111a8c16d67c81d](#)

Direct internet access

Enabled: [Learn more](#)

4. Start the AWS Notebook.

The screenshot shows the AWS SageMaker console. The left sidebar contains the 'Amazon SageMaker' menu with options: Getting started, Studio, Studio Lab, Canvas, and RStudio. The main content area is titled 'Amazon SageMaker > Notebook instances'. It features a 'Notebook instances' section with a search bar and a 'Create notebook instance' button. Below this is a table with the following data:

Name	Instance	Creation time	Status	Actions
nkarthiksagemaker	ml.t2.medium	2/16/2023, 1:55:38 PM	InService	Open Jupyter Open JupyterLab

5. Open the Jupyter lab.



6. Log into the terminal and mount the Cloud Volumes ONTAP volume.

```
sh-4.2$ sudo mkdir /vol1; sudo mount -t nfs 172.30.10.41:/vol1 /vol1
sh-4.2$ df -h
```

Filesystem	Size	Used	Avail	Use%	Mounted on
devtmpfs	2.0G	0	2.0G	0%	/dev
tmpfs	2.0G	0	2.0G	0%	/dev/shm
tmpfs	2.0G	624K	2.0G	1%	/run
tmpfs	2.0G	0	2.0G	0%	/sys/fs/cgroup
/dev/xvda1	140G	114G	27G	82%	/
/dev/xvdf	4.8G	72K	4.6G	1%	/home/ec2-user/SageMaker
tmpfs	393M	0	393M	0%	/run/user/1001
tmpfs	393M	0	393M	0%	/run/user/1002
tmpfs	393M	0	393M	0%	/run/user/1000
172.30.10.41:/vol1	973M	189M	785M	20%	/vol1

```
sh-4.2$
```

7. Check the bucket created on the Cloud Volumes ONTAP volume using the AWS CLI commands.

```
sh-4.2$ aws configure --profile netapp
AWS Access Key ID [None]: 0ZNAX21JW5Q8AP80CQ2E
AWS Secret Access Key [None]: PpLs4gA9K0_2gPhuykkp014gBjcC9Rbi3QDX_6rr
Default region name [None]: us-east-1
Default output format [None]:
sh-4.2$

sh-4.2$ aws s3 ls --profile netapp --endpoint-url
2023-02-10 17:59:48 ontapbucket1

sh-4.2$ aws s3 ls --profile netapp --endpoint-url s3://ontapbucket1/

2023-02-10 18:46:44          4747 1
2023-02-10 18:48:32          96 setup.cfg

sh-4.2$
```

Data for machine learning

In this validation, we used a dataset from DBpedia, a crowd-sourced community effort, to extract structured content from the information created in various Wikimedia projects.

1. Download the data from the DBpedia GitHub location and extract it. Use the same terminal used in the previous section.

```

sh-4.2$ wget
--2023-02-14 23:12:11--
Resolving github.com (github.com)... 140.82.113.3
Connecting to github.com (github.com)|140.82.113.3|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: [following]
--2023-02-14 23:12:11--
Resolving raw.githubusercontent.com (raw.githubusercontent.com)...
185.199.109.133, 185.199.110.133, 185.199.111.133, ...
Connecting to raw.githubusercontent.com
(raw.githubusercontent.com)|185.199.109.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 68431223 (65M) [application/octet-stream]
Saving to: 'dbpedia_csv.tar.gz'

100%[=====
=====
=====>] 68,431,223  56.2MB/s   in 1.2s

2023-02-14 23:12:13 (56.2 MB/s) - 'dbpedia_csv.tar.gz' saved
[68431223/68431223]

sh-4.2$ tar -zxvf dbpedia_csv.tar.gz
dbpedia_csv/
dbpedia_csv/test.csv
dbpedia_csv/classes.txt
dbpedia_csv/train.csv
dbpedia_csv/readme.txt
sh-4.2$

```

2. Copy the data to the Cloud Volumes ONTAP location and check it from the S3 bucket using the AWS CLI.


```

sh-4.2$ df -h
Filesystem                Size      Used Avail Use% Mounted on
devtmpfs                  2.0G         0   2.0G   0% /dev
tmpfs                     2.0G         0   2.0G   0% /dev/shm
tmpfs                     2.0G    628K   2.0G   1% /run
tmpfs                     2.0G         0   2.0G   0% /sys/fs/cgroup
/dev/xvda1                140G    114G   27G   82% /
/dev/xvdf                 4.8G     52K   4.6G   1% /home/ec2-user/SageMaker
tmpfs                     393M         0   393M   0% /run/user/1002
tmpfs                     393M         0   393M   0% /run/user/1001
tmpfs                     393M         0   393M   0% /run/user/1000
172.30.10.41:/vol1        973M    384K   973M   1% /vol1
sh-4.2$ pwd
/home/ec2-user
sh-4.2$ cp -ra dbpedia_csv /vol1
sh-4.2$ aws s3 ls --profile netapp --endpoint-url s3://ontapbucket1/
                PRE dbpedia_csv/
2023-02-10 18:46:44          4747 1
2023-02-10 18:48:32           96 setup.cfg
sh-4.2$

```

3. Perform basic validation to make sure that read/write functionality works on the S3 bucket.

```

sh-4.2$ aws s3 cp --profile netapp --endpoint-url /usr/share/doc/util-
linux-2.30.2 s3://ontapbucket1/ --recursive
upload: ../../usr/share/doc/util-linux-2.30.2/deprecated.txt to
s3://ontapbucket1/deprecated.txt
upload: ../../usr/share/doc/util-linux-2.30.2/getopt-parse.bash to
s3://ontapbucket1/getopt-parse.bash
upload: ../../usr/share/doc/util-linux-2.30.2/README to
s3://ontapbucket1/README
upload: ../../usr/share/doc/util-linux-2.30.2/getopt-parse.tcsh to
s3://ontapbucket1/getopt-parse.tcsh
upload: ../../usr/share/doc/util-linux-2.30.2/AUTHORS to
s3://ontapbucket1/AUTHORS
upload: ../../usr/share/doc/util-linux-2.30.2/NEWS to
s3://ontapbucket1/NEWS
sh-4.2$ aws s3 ls --profile netapp --endpoint-url
s3://ontapbucket1/s3://ontapbucket1/

An error occurred (InternalError) when calling the ListObjectsV2
operation: We encountered an internal error. Please try again.
sh-4.2$ aws s3 ls --profile netapp --endpoint-url s3://ontapbucket1/
                PRE dbpedia_csv/
2023-02-16 19:19:27        26774 AUTHORS

```

```

2023-02-16 19:19:27      72727 NEWS
2023-02-16 19:19:27      4493 README
2023-02-16 19:19:27      2825 deprecated.txt
2023-02-16 19:19:27      1590 getopt-parse.bash
2023-02-16 19:19:27      2245 getopt-parse.tcsh
sh-4.2$ ls -ltr /vol1
total 132
drwxrwxr-x 2 ec2-user ec2-user 4096 Mar 29 2015 dbpedia_csv
-rw-r--r-- 1 nobody  nobody  2245 Apr 10 17:37 getopt-parse.tcsh
-rw-r--r-- 1 nobody  nobody  2825 Apr 10 17:37 deprecated.txt
-rw-r--r-- 1 nobody  nobody  4493 Apr 10 17:37 README
-rw-r--r-- 1 nobody  nobody  1590 Apr 10 17:37 getopt-parse.bash
-rw-r--r-- 1 nobody  nobody 26774 Apr 10 17:37 AUTHORS
-rw-r--r-- 1 nobody  nobody 72727 Apr 10 17:37 NEWS
sh-4.2$ ls -ltr /vol1/dbpedia_csv/
total 192104
-rw----- 1 ec2-user ec2-user 174148970 Mar 28 2015 train.csv
-rw----- 1 ec2-user ec2-user 21775285 Mar 28 2015 test.csv
-rw----- 1 ec2-user ec2-user      146 Mar 28 2015 classes.txt
-rw-rw-r-- 1 ec2-user ec2-user      1758 Mar 29 2015 readme.txt
sh-4.2$ chmod -R 777 /vol1/dbpedia_csv
sh-4.2$ ls -ltr /vol1/dbpedia_csv/
total 192104
-rwxrwxrwx 1 ec2-user ec2-user 174148970 Mar 28 2015 train.csv
-rwxrwxrwx 1 ec2-user ec2-user 21775285 Mar 28 2015 test.csv
-rwxrwxrwx 1 ec2-user ec2-user      146 Mar 28 2015 classes.txt
-rwxrwxrwx 1 ec2-user ec2-user      1758 Mar 29 2015 readme.txt
sh-4.2$ aws s3 cp --profile netapp --endpoint-url http://172.30.2.248/
s3://ontapbucket1/ /tmp --recursive
download: s3://ontapbucket1/AUTHORS to ../../tmp/AUTHORS
download: s3://ontapbucket1/README to ../../tmp/README
download: s3://ontapbucket1/NEWS to ../../tmp/NEWS
download: s3://ontapbucket1/dbpedia_csv/classes.txt to
../../tmp/dbpedia_csv/classes.txt
download: s3://ontapbucket1/dbpedia_csv/readme.txt to
../../tmp/dbpedia_csv/readme.txt
download: s3://ontapbucket1/deprecated.txt to ../../tmp/deprecated.txt
download: s3://ontapbucket1/getopt-parse.bash to ../../tmp/getopt-
parse.bash
download: s3://ontapbucket1/getopt-parse.tcsh to ../../tmp/getopt-
parse.tcsh
download: s3://ontapbucket1/dbpedia_csv/test.csv to
../../tmp/dbpedia_csv/test.csv
download: s3://ontapbucket1/dbpedia_csv/train.csv to
../../tmp/dbpedia_csv/train.csv
sh-4.2$

```

```
sh-4.2$ aws s3 ls --profile netapp --endpoint-url s3://ontapbucket1/
                PRE dbpedia_csv/
2023-02-16 19:19:27      26774 AUTHORS
2023-02-16 19:19:27      72727 NEWS
2023-02-16 19:19:27      4493 README
2023-02-16 19:19:27      2825 deprecated.txt
2023-02-16 19:19:27      1590 getopt-parse.bash
2023-02-16 19:19:27      2245 getopt-parse.tcsh
sh-4.2$
```

Validate machine learning from Jupyter Notebooks

The following validation provides the machine-learning build, train, and deploy models through text classification by using the SageMaker BlazingText example below:

1. Install the boto3 and SageMaker packages.

```
In [1]: pip install --upgrade boto3 sagemaker
```

Output:

```
Looking in indexes: https://pypi.org/simple,
https://pip.repos.neuron.amazonaws.com
Requirement already satisfied: boto3 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (1.26.44)
Collecting boto3
  Downloading boto3-1.26.72-py3-none-any.whl (132 kB)
    132.7/132.7 kB 14.6 MB/s eta
0: 00:00
Requirement already satisfied: sagemaker in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (2.127.0)
Collecting sagemaker
  Downloading sagemaker-2.132.0.tar.gz (668 kB)
    668.0/668.0 kB 12.3 MB/s eta
0:
00:0000:01
  Preparing metadata (setup.py) ... done
Collecting botocore<1.30.0,>=1.29.72
  Downloading botocore-1.29.72-py3-none-any.whl (10.4 MB)
    10.4/10.4 MB 44.3 MB/s eta
0: 00:0000:010:01
Requirement already satisfied: s3transfer<0.7.0,>=0.6.0 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from boto3)
(0.6.0)
Requirement already satisfied: jmespath<2.0.0,>=0.7.1 in /home/ec2-
```

```

user/anaconda3/envs/python3/lib/python3.10/site-packages (from boto3)
(0.10.0)
Requirement already satisfied: attrs<23,>=20.3.0 in /home/ec2-
user/anaconda
3/envs/python3/lib/python3.10/site-packages (from sagemaker) (22.1.0)
Requirement already satisfied: google-pasta in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker) (0.2.0)
Requirement already satisfied: numpy<2.0,>=1.9.0 in /home/ec2-
user/anaconda
3/envs/python3/lib/python3.10/site-packages (from sagemaker) (1.22.4)
Requirement already satisfied: protobuf<4.0,>=3.1 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker) (3.20.3)
Requirement already satisfied: protobuf3-to-dict<1.0,>=0.1.5 in
/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages
(from sagemaker)
(0.1.5)
Requirement already satisfied: smdebug_rulesconfig==1.0.1 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker) (1.
0.1) Requirement already satisfied: importlib-metadata<5.0,>=1.4.0 in
/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker)
(4.13.0)
Requirement already satisfied: packaging>=20.0 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker) (21.3)
Requirement already satisfied: pandas in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker) (1.5.1)
Requirement already satisfied: pathos in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker) (0.3.0)
Requirement already satisfied: schema in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
sagemaker) (0.7.5) Requirement already satisfied: python-
dateutil<3.0.0,>=2.1 in /home/ec2-user
r/anaconda3/envs/python3/lib/python3.10/site-packages (from
botocore<1.30.
0,>=1.29.72->boto3) (2.8.2)
Requirement already satisfied: urllib3<1.27,>=1.25.4 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
botocore<1.30.0,>=1.2
9.72->boto3) (1.26.8) Requirement already satisfied: zipp>=0.5 in
/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages

```

```

(from importlib-metadata<5.0,>=1.4.0->sagemaker) (3.10.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from
packaging>=20.0->sagemaker) (3.0.9)
Requirement already satisfied: six in /home/ec2-
user/anaconda3/envs/python
3/lib/python3.10/site-packages (from protobuf3-to-dict<1.0,>=0.1.5-
>sagemaker) (1.16.0)
Requirement already satisfied: pytz>=2020.1 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from pandas-
>sagemaker) (2022.5)
Requirement already satisfied: ppft>=1.7.6.6 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from pathos-
>sagemaker) (1.7.6.6) Requirement already satisfied:
multiprocess>=0.70.14 in /home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages (from pathos->sagemaker)
(0.70.14)
Requirement already satisfied: dill>=0.3.6 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from pathos-
>sagemaker) (0.3.6)
Requirement already satisfied: pox>=0.3.2 in /home/ec2-
user/anaconda3/envs/python3/lib/python3.10/site-packages (from pathos-
>sagemaker) (0.3.2) Requirement already satisfied: contextlib2>=0.5.5 in
/home/ec2-user/anaconda3/envs/python3/lib/python3.10/site-packages
(from schema->sagemaker) (21.
6.0) Building wheels for collected packages: sagemaker
  Building wheel for sagemaker (setup.py) ... done
  Created wheel for sagemaker: filename=sagemaker-2.132.0-py2.py3-none-
any.whl size=905449
sha256=f6100a5dc95627f2e2a49824e38f0481459a27805ee19b5a06ec
83db0252fd41
  Stored in directory: /home/ec2-
user/.cache/pip/wheels/60/41/b6/482e7ab096
520df034fbf2d44a1d7ba0681b27ef45aa61
Successfully built sagemaker
Installing collected packages: botocore, boto3, sagemaker
  Attempting uninstall: botocore      Found existing installation:
botocore 1.24.19
    Uninstalling botocore-1.24.19:      Successfully uninstalled
botocore-1.24.19
  Attempting uninstall: boto3      Found existing installation: boto3
1.26.44
    Uninstalling boto3-1.26.44:
      Successfully uninstalled boto3-1.26.44
  Attempting uninstall: sagemaker      Found existing installation:
sagemaker 2.127.0

```

```
Uninstalling sagemaker-2.127.0:
```

```
Successfully uninstalled sagemaker-2.127.0
```

```
ERROR: pip's dependency resolver does not currently take into account  
all the packages that are installed. This behaviour is the source of  
the following dependency conflicts.
```

```
awscli 1.27.44 requires botocore==1.29.44, but you have botocore 1.29.72  
which is incompatible.
```

```
aiobotocore 2.0.1 requires botocore<1.22.9,>=1.22.8, but you have  
botocore 1.29.72 which is incompatible. Successfully installed boto3-
```

```
1.26.72 botocore-1.29.72 sagemaker-2.132.0 Note: you may need to restart  
the kernel to use updated packages.
```

2. In the following step, the data (dbpedia_csv) is downloaded from the s3 bucket `ontapbucket1` to a Jupyter Notebook instance used in machine learning.

```

In [2]: import sagemaker
In [3]: from sagemaker import get_execution_role
In [4]:
import json
import boto3
sess = sagemaker.Session()
role = get_execution_role()
print(role)
bucket = "ontapbucket1"
print(bucket)
sess.s3_client = boto3.client('s3',region_name='',aws_access_key_id =
'0ZNAX21JW5Q8AP80CQ2E', aws_secret_access_key =
'PpLs4gA9K0_2gPhuykkp014gBjcC9Rbi3QDX_6rr',
                                use_ssl = False, endpoint_url =
'http://172.30.10.41',

config=boto3.session.Config(signature_version='s3v4',
s3={'addressing_style':'path'}) )
sess.s3_resource = boto3.resource('s3',region_name='',aws_access_key_id
= '0ZNAX21JW5Q8AP80CQ2E', aws_secret_access_key =
'PpLs4gA9K0_2gPhuykkp014gBjcC9Rbi3QDX_6rr',
                                use_ssl = False, endpoint_url =
'http://172.30.10.41',

config=boto3.session.Config(signature_version='s3v4',
s3={'addressing_style':'path'}) )
prefix = "blazingtext/supervised"
import os
my_bucket = sess.s3_resource.Bucket(bucket)
my_bucket = sess.s3_resource.Bucket(bucket)
#os.mkdir('dbpedia_csv')
for s3_object in my_bucket.objects.all():
    filename = s3_object.key
#    print(filename)
#    print(s3_object.key)
    my_bucket.download_file(s3_object.key, filename)

```

3. The following code creates the mapping from integer indices to class labels that are used to retrieve the actual class name during inference.

```

index_to_label = {}
with open("dbpedia_csv/classes.txt") as f:
    for i,label in enumerate(f.readlines()):
        index_to_label[str(i + 1)] = label.strip()

```

The output lists the files and folders in the `ontapbucket1` bucket that are used as data for the AWS SageMaker machine-learning validation.

```
arn:aws:iam::210811600188:role/SageMakerFullRole ontapbucket1
AUTHORS
AUTHORS
NEWS
NEWS
README README
dbpedia_csv/classes.txt dbpedia_csv/classes.txt dbpedia_csv/readme.txt
dbpedia_csv/readme.txt dbpedia_csv/test.csv dbpedia_csv/test.csv
dbpedia_csv/train.csv dbpedia_csv/train.csv deprecated.txt
deprecated.txt getopt-parse.bash getopt-parse.bash getopt-parse.tcsh
getopt-parse.tcsh
In [5]: ls
AUTHORS          deprecated.txt    getopt-parse.tcsh NEWS
Untitled.ipynb dbpedia_csv/    getopt-parse.bash lost+found/
README
In [6]: ls -l dbpedia_csv
total 191344
-rw-rw-r-- 1 ec2-user ec2-user      146 Feb 16 19:43 classes.txt
-rw-rw-r-- 1 ec2-user ec2-user     1758 Feb 16 19:43 readme.txt
-rw-rw-r-- 1 ec2-user ec2-user  21775285 Feb 16 19:43 test.csv
-rw-rw-r-- 1 ec2-user ec2-user 174148970 Feb 16 19:43 train.csv
```

4. Start the data preprocessing phase to preprocess the training data into a space-separated, tokenized text format that can be consumed by the BlazingText algorithm and the `nlTK` library to tokenize the input sentences from the DBPedia dataset. Download the `nlTK` tokenizer and other libraries. The `transform_instance` applied to each data instance in parallel uses the Python multiprocessing module.

```
In [7]: from random import shuffle
import multiprocessing
from multiprocessing import Pool
import csv
import nltk
nltk.download("punkt")
def transform_instance(row):
    cur_row = []
    label = "__label__" + index_to_label [row[0]] # Prefix the index-ed
label with __label__
    cur_row.append (label)
    cur_row.extend(nltk.word_tokenize(row[1].lower ()))
    cur_row.extend(nltk.word_tokenize(row[2].lower ()))
    return cur_row
def preprocess(input_file, output_file, keep=1):
```



```

all_rows = []
with open(input_file,"r") as csvinfile:
    csv_reader = csv.reader(csvinfile, delimiter=",")
    for row in csv_reader:
        all_rows.append(row)
shuffle(all_rows)
all_rows = all_rows[: int(keep * len(all_rows))]
pool = Pool(processes=multiprocessing.cpu_count())
transformed_rows = pool.map(transform_instance, all_rows)
pool.close()
pool.join()
with open(output_file, "w") as csvoutfile:
    csv_writer = csv.writer (csvoutfile, delimiter=" ",
lineterminator="\n")
    csv_writer.writerows (transformed_rows)

# Preparing the training dataset
# since preprocessing the whole dataset might take a couple of minutes,
# we keep 20% of the training dataset for this demo.
# Set keep to 1 if you want to use the complete dataset
preprocess("dbpedia_csv/train.csv","dbpedia.train", keep=0.2)
# Preparing the validation dataset
preprocess("dbpedia_csv/test.csv","dbpedia.validation")
sess = sagemaker.Session()
role = get_execution_role()
print (role) # This is the role that sageMaker would use to leverage Aws
resources (S3, Cloudwatch) on your behalf
bucket = sess.default_bucket() # Replace with your own bucket name if
needed
print("default Bucket:: ")
print(bucket)

```

Output:

```

[nltk_data] Downloading package punkt to /home/ec2-user/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
arn:aws:iam::210811600188:role/SageMakerFullRole default Bucket::
sagemaker-us-east-1-210811600188

```

5. Upload the formatted and training dataset to S3 so that it can be used by SageMaker to execute training jobs. Then upload two files to the bucket and prefix location using the Python SDK.

```

In [8]: %%time
train_channel = prefix + "/train"
validation_channel = prefix + "/validation"
sess.upload_data(path="dbpedia.train", bucket=bucket,
key_prefix=train_channel)
sess.upload_data(path="dbpedia.validation", bucket=bucket,
key_prefix=validation_channel)
s3_train_data = "s3://{}/{}".format(bucket, train_channel)
s3_validation_data = "s3://{}/{}".format(bucket, validation_channel)

```

Output:

```

CPU times: user 546 ms, sys: 163 ms, total: 709 ms
Wall time: 1.32 s

```

6. Set up an output location at S3 where the model artifact is loaded so that artifacts can be the output of the algorithm's training job. Create a `sageMaker.estimator.Estimator` object to launch the training job.

```

In [9]: s3_output_location = "s3://{}/output".format(bucket, prefix)
In [10]: region_name = boto3.Session().region_name
In [11]: container =
sagemaker.amazon.amazon_estimator.get_image_uri(region_name,
"blazingtext", "latest")
print("Using SageMaker BlazingText container: {} ({}).format(container,
region_name))

```

Output:

```

The method get_image_uri has been renamed in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
Defaulting to the only supported framework/algorithm version: 1.
Ignoring framework/algorithm version: latest.
Using SageMaker BlazingText container: 811284229777.dkr.ecr.us-east-1.amazonaws.com/blazingtext:1 (us-east-1)

```

7. Define the `SageMaker Estimator` with resource configurations and hyperparameters to train text classification on the DBPedia dataset using the supervised mode on a `c4.4xlarge` instance.

```

In [12]: bt_model = sagemaker.estimator.Estimator(
    container,
    role,
    instance_count=1,
    instance_type="ml.c4.4xlarge",
    volume_size=30,
    max_run=360000,
    input_mode="File",
    output_path=s3_output_location,
    hyperparameters={
        "mode": "supervised",
        "epochs": 1,
        "min_count": 2,
        "learning_rate": 0.05,
        "vector_dim": 10,
        "early_stopping": True,
        "patience": 4,
        "min_epochs": 5,
        "word_ngrams": 2,
    },
)

```

8. Prepare a handshake between the data channels and the algorithm. To do this, create the `sagemaker.session.s3_input` objects from the data channels, and keep them in a dictionary for the algorithm to consume.

```

In [13]: train_data = sagemaker.inputs.TrainingInput(
    s3_train_data,
    distribution="FullyReplicated",
    content_type="text/plain",
    s3_data_type="S3Prefix",
)
validation_data = sagemaker.inputs.TrainingInput(
    s3_validation_data,
    distribution="FullyReplicated",
    content_type="text/plain",
    s3_data_type="S3Prefix",
)
data_channels = {"train": train_data, "validation": validation_data}

```

9. After the job has finished, a Job Complete message appears. The trained model can be found in the S3 bucket that was set up as the `output_path` in the estimator.

```
ln [14]: bt_model.fit(inputs=data_channels, logs=True)
```

Output:

```
INFO:sagemaker:Creating training-job with name: blazingtext-2023-02-16-
20-3
7-30-748
2023-02-16 20:37:30 Starting - Starting the training job.....
2023-02-16 20:38:09 Starting - Preparing the instances for
training.....
2023-02-16 20:39:24 Downloading - Downloading input data
2023-02-16 20:39:24 Training - Training image download completed.
Training in progress... Arguments: train
[02/16/2023 20:39:41 WARNING 140279908747072] Loggers have already been
set up. [02/16/2023 20:39:41 WARNING 140279908747072] Loggers have
already been set up.
[02/16/2023 20:39:41 INFO 140279908747072] nvidia-smi took:
0.0251793861389
16016 secs to identify 0 gpus
[02/16/2023 20:39:41 INFO 140279908747072] Running single machine CPU
BlazingText training using supervised mode.
Number of CPU sockets found in instance is 1
[02/16/2023 20:39:41 INFO 140279908747072] Processing
/opt/ml/input/data/training/dbpedia.train . File size: 35.0693244934082 MB
[02/16/2023 20:39:41 INFO 140279908747072] Processing
/opt/ml/input/data/validation/dbpedia.validation . File size:
21.887572288513184 MB
Read 6M words
Number of words: 149301
Loading validation data from
/opt/ml/input/data/validation/dbpedia.validation
Loaded validation data.
----- End of epoch: 1 ##### Alpha: 0.0000 Progress: 100.00%
Million Words/sec: 10.39 ##### Training finished.
Average throughput in Million words/sec: 10.39
Total training time in seconds: 0.60
#train_accuracy: 0.7223
Number of train examples: 112000
#validation_accuracy: 0.7205
Number of validation examples: 70000
2023-02-16 20:39:55 Uploading - Uploading generated training model
2023-02-16 20:40:11 Completed - Training job completed
Training seconds: 68
Billable seconds: 68
```

10. After training is complete, deploy the trained model as an Amazon SageMaker real-time hosted endpoint to make predictions.

```
In [15]: from sagemaker.serializers import JSONSerializer
text_classifier = bt_model.deploy(
    initial_instance_count=1, instance_type="ml.m4.xlarge",
    serializer=JSONS
)
```

Output:

```
INFO:sagemaker:Creating model with name: blazingtext-2023-02-16-20-41-33-10
0
INFO:sagemaker:Creating endpoint-config with name blazingtext-2023-02-16-20-41-33-100
INFO:sagemaker:Creating endpoint with name blazingtext-2023-02-16-20-41-33-100
-----!
```

```
In [16]: sentences = [
    "Convair was an american aircraft manufacturing company which later expanded into rockets and spacecraft.",
    "Berwick secondary college is situated in the outer melbourne metropolitan suburb of berwick .",
]
# using the same nltk tokenizer that we used during data preparation for training
tokenized_sentences = [" ".join(nltk.word_tokenize(sent)) for sent in sentences]
payload = {"instances": tokenized_sentences} response = text_classifier.predict(payload)
predictions = json.loads(response)
print(json.dumps(predictions, indent=2))
```

```
[
  {
    "label": [
      "__label__Artist"
    ],
    "prob": [
      0.4090951681137085
    ]
  },
  {
    "label": [
      "__label__EducationalInstitution"
    ],
    "prob": [
      0.49466073513031006
    ]
  }
]
```

11. By default, the model returns one prediction with the highest probability. To retrieve the top k predictions, set k in the configuration file.

```
In [17]: payload = {"instances": tokenized_sentences, "configuration":
{"k": 2}}
response = text_classifier.predict(payload)

predictions = json.loads(response)
print(json.dumps(predictions, indent=2))
```

```
[
  {
    "label": [
      "__label__Artist",
      "__label__MeanOfTransportation"
    ],
    "prob": [
      0.4090951681137085,
      0.26930734515190125
    ]
  },
  {
    "label": [
      "__label__EducationalInstitution",
      "__label__Building"
    ],
    "prob": [
      0.49466073513031006,
      0.15817692875862122
    ]
  }
]
```

12. Delete the endpoint before closing the notebook.

```
In [18]: sess.delete_endpoint(text_classifier.endpoint)
WARNING:sagemaker.deprecations:The endpoint attribute has been renamed
in sagemaker>=2.
See: https://sagemaker.readthedocs.io/en/stable/v2.html for details.
INFO:sagemaker:Deleting endpoint with name: blazingtext-2023-02-16-20-
41-33
-100
```

[Next: Conclusion.](#)

Conclusion

[Previous: Data duality for data scientists and other applications.](#)

Based on this validation, Data scientists and engineers can access NFS data from AWS SageMaker Jupyter Notebooks via S3 buckets from NetApp Cloud Volumes ONTAP. This approach enables easy access and sharing of the same data from both NFS and S3 without the need for additional software.

Where to find additional information

To learn more about the information that is described in this document, review the following documents and/or websites:

- Text classification using SageMaker BlazingText

https://sagemaker-examples.readthedocs.io/en/latest/introduction_to_amazon_algorithms/blazingtext_text_classification_dbpedia/blazingtext_text_classification_dbpedia.html

- ONTAP version support for S3 object storage

<https://docs.netapp.com/us-en/ontap/s3-config/ontap-version-support-s3-concept.html>

Version history

Version	Date	Document version history
Version 1.0	April 2023	Initial release

Copyright information

Copyright © 2023 NetApp, Inc. All Rights Reserved. Printed in the U.S. No part of this document covered by copyright may be reproduced in any form or by any means—graphic, electronic, or mechanical, including photocopying, recording, taping, or storage in an electronic retrieval system—without prior written permission of the copyright owner.

Software derived from copyrighted NetApp material is subject to the following license and disclaimer:

THIS SOFTWARE IS PROVIDED BY NETAPP “AS IS” AND WITHOUT ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE, WHICH ARE HEREBY DISCLAIMED. IN NO EVENT SHALL NETAPP BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

NetApp reserves the right to change any products described herein at any time, and without notice. NetApp assumes no responsibility or liability arising from the use of products described herein, except as expressly agreed to in writing by NetApp. The use or purchase of this product does not convey a license under any patent rights, trademark rights, or any other intellectual property rights of NetApp.

The product described in this manual may be protected by one or more U.S. patents, foreign patents, or pending applications.

LIMITED RIGHTS LEGEND: Use, duplication, or disclosure by the government is subject to restrictions as set forth in subparagraph (b)(3) of the Rights in Technical Data -Noncommercial Items at DFARS 252.227-7013 (FEB 2014) and FAR 52.227-19 (DEC 2007).

Data contained herein pertains to a commercial product and/or commercial service (as defined in FAR 2.101) and is proprietary to NetApp, Inc. All NetApp technical data and computer software provided under this Agreement is commercial in nature and developed solely at private expense. The U.S. Government has a non-exclusive, non-transferrable, nonsublicensable, worldwide, limited irrevocable license to use the Data only in connection with and in support of the U.S. Government contract under which the Data was delivered. Except as provided herein, the Data may not be used, disclosed, reproduced, modified, performed, or displayed without the prior written approval of NetApp, Inc. United States Government license rights for the Department of Defense are limited to those rights identified in DFARS clause 252.227-7015(b) (FEB 2014).

Trademark information

NETAPP, the NETAPP logo, and the marks listed at <http://www.netapp.com/TM> are trademarks of NetApp, Inc. Other company and product names may be trademarks of their respective owners.