
AWS
re:Invent

SVS301

Analyzing data at any scale with AWS Lambda

Danilo Poccia

Chief Evangelist (EMEA)
AWS

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

Event-driven metadata extraction and enrichment

Monte Carlo method, ensemble forecasting, ensemble learning

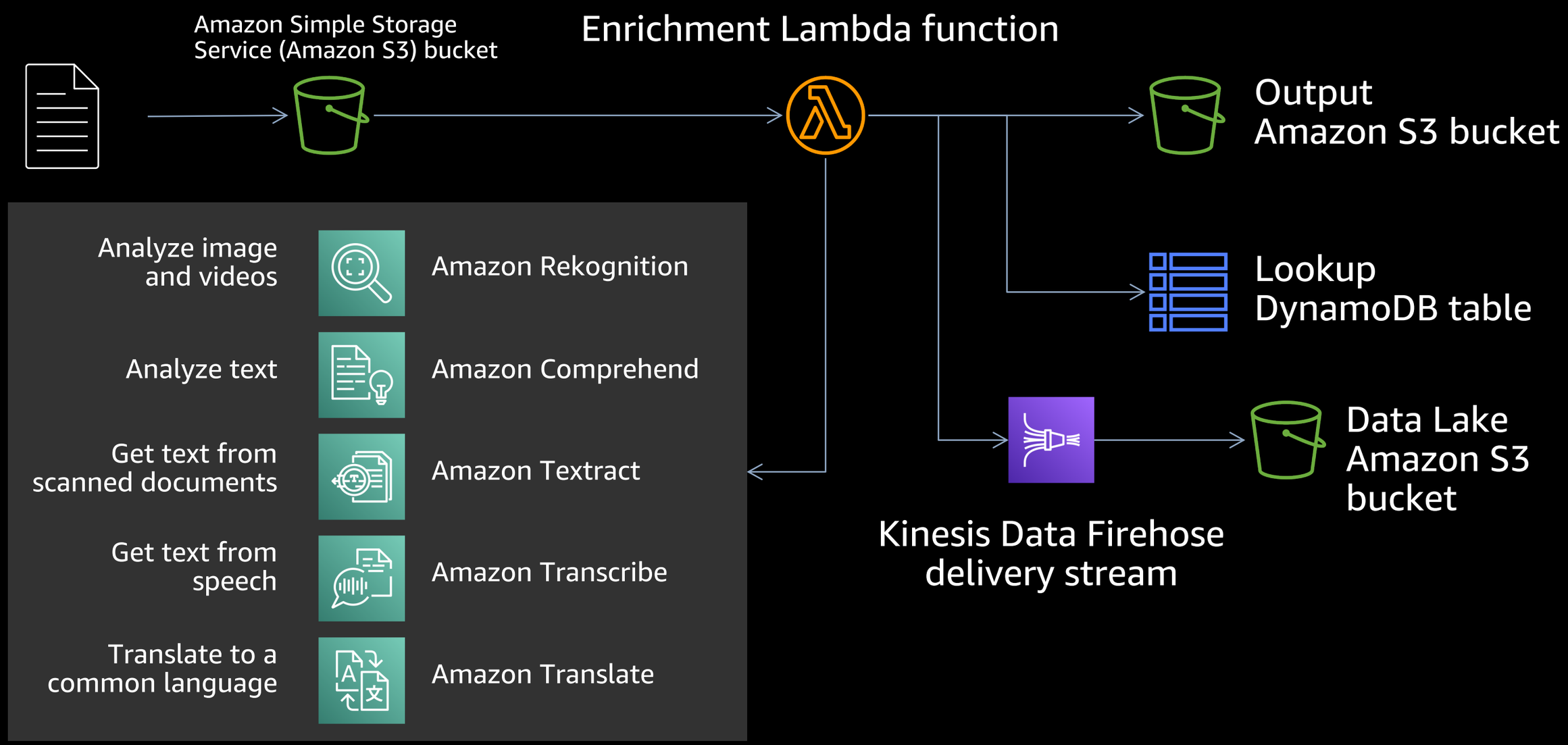
Using Amazon Elastic File System (Amazon EFS) to host Lambda function dependencies

Machine learning inference examples

Advanced use cases and demos

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Event-driven metadata extraction + enrichment



Monte Carlo methods

REPEATED RANDOM SAMPLING AND STATISTICAL ANALYSIS

Physical sciences and engineering

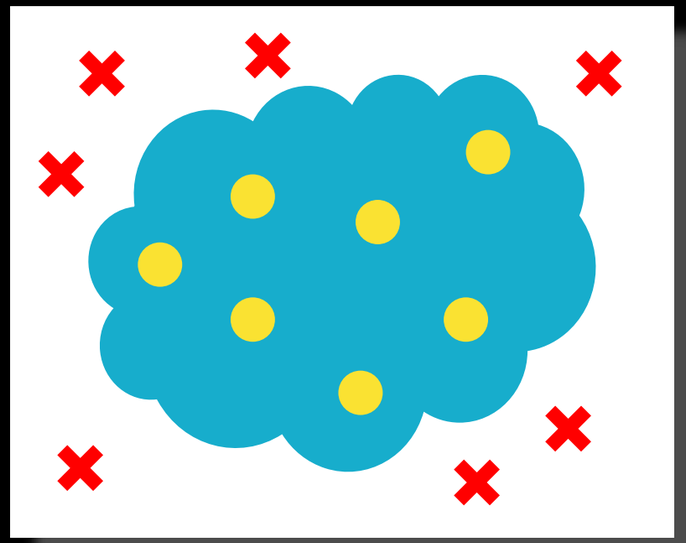
Computational physics, physical chemistry, aerodynamics, fluid dynamics, wireless networks

Finance and business

Evaluate the risk and uncertainty that would affect the outcome of different decision options

Mathematics

For example, execute 10 / 100 / 1,000 Lambda functions, each performing one or more random sampling



What's the area of the "cloud" shape?

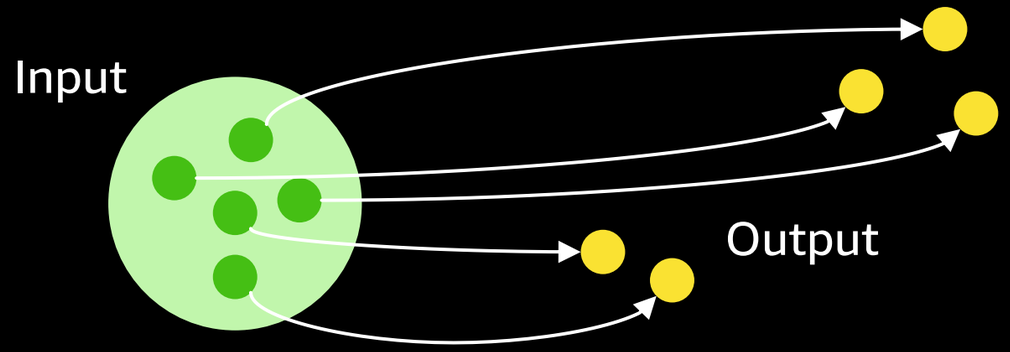
How many hits and misses?

The number of hits divided by the total number of runs tends to the same ratio as the two areas

Ensemble forecasting

INSTEAD OF MAKING A SINGLE FORECAST OF THE MOST LIKELY SCENARIO, A SET (OR ENSEMBLE) OF FORECASTS IS PRODUCED

This set of forecasts aims to give a better indication of the range of possible future states



Useful when there is uncertainty/error in the input parameters, or a system is highly sensitive to initial conditions – such as for chaotic dynamical systems, like weather forecast

For example, execute 10 / 100 / 1,000 Lambda functions, each with slightly different parameters in input, and then compare the results

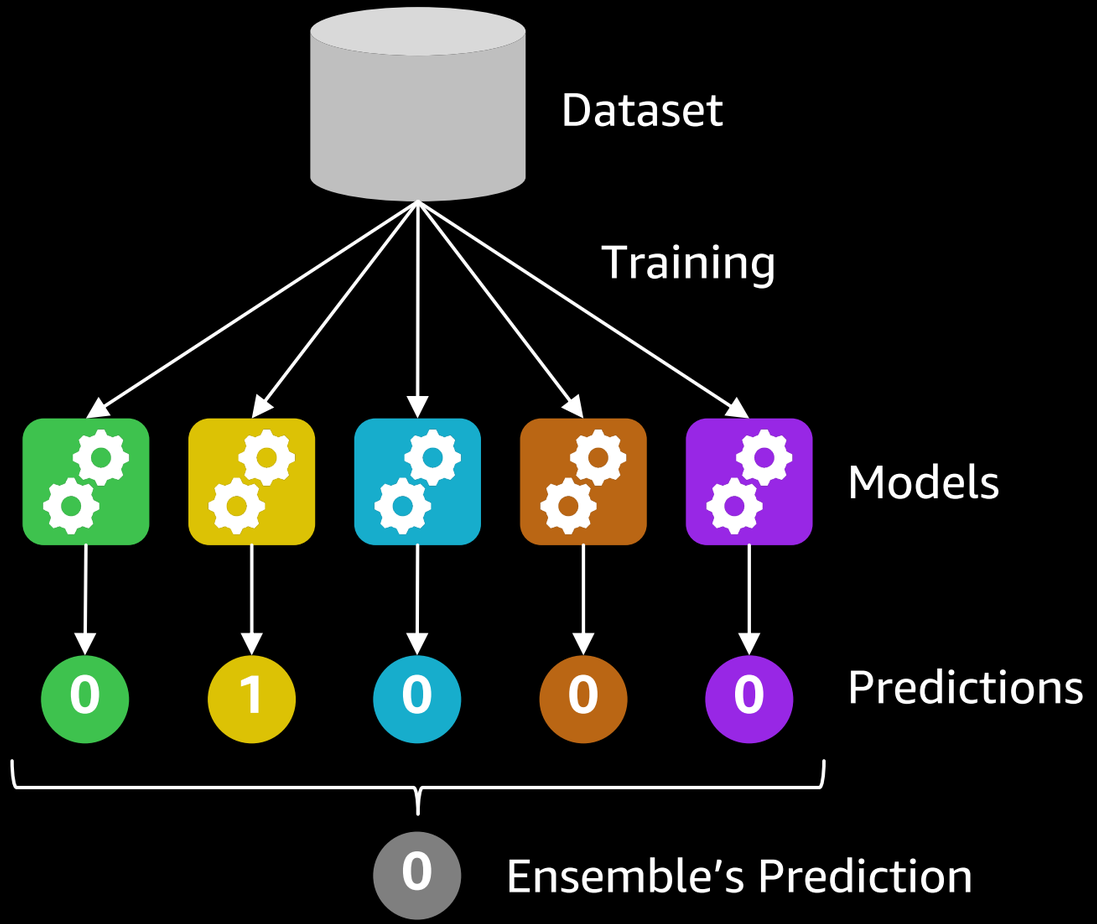
If 56 out of 100 weather forecast simulations expect rain, then, “There is a 56% chance of rain”

Ensemble learning

USING MULTIPLE LEARNING ALGORITHMS TO OBTAIN BETTER PREDICTIVE PERFORMANCE
THAN WHAT COULD BE OBTAINED FROM ANY OF THE CONSTITUENT LEARNING ALGORITHMS

For example, run 10 / 100 / 1,000 Lambda functions training different (and relatively simple) machine learning algorithms

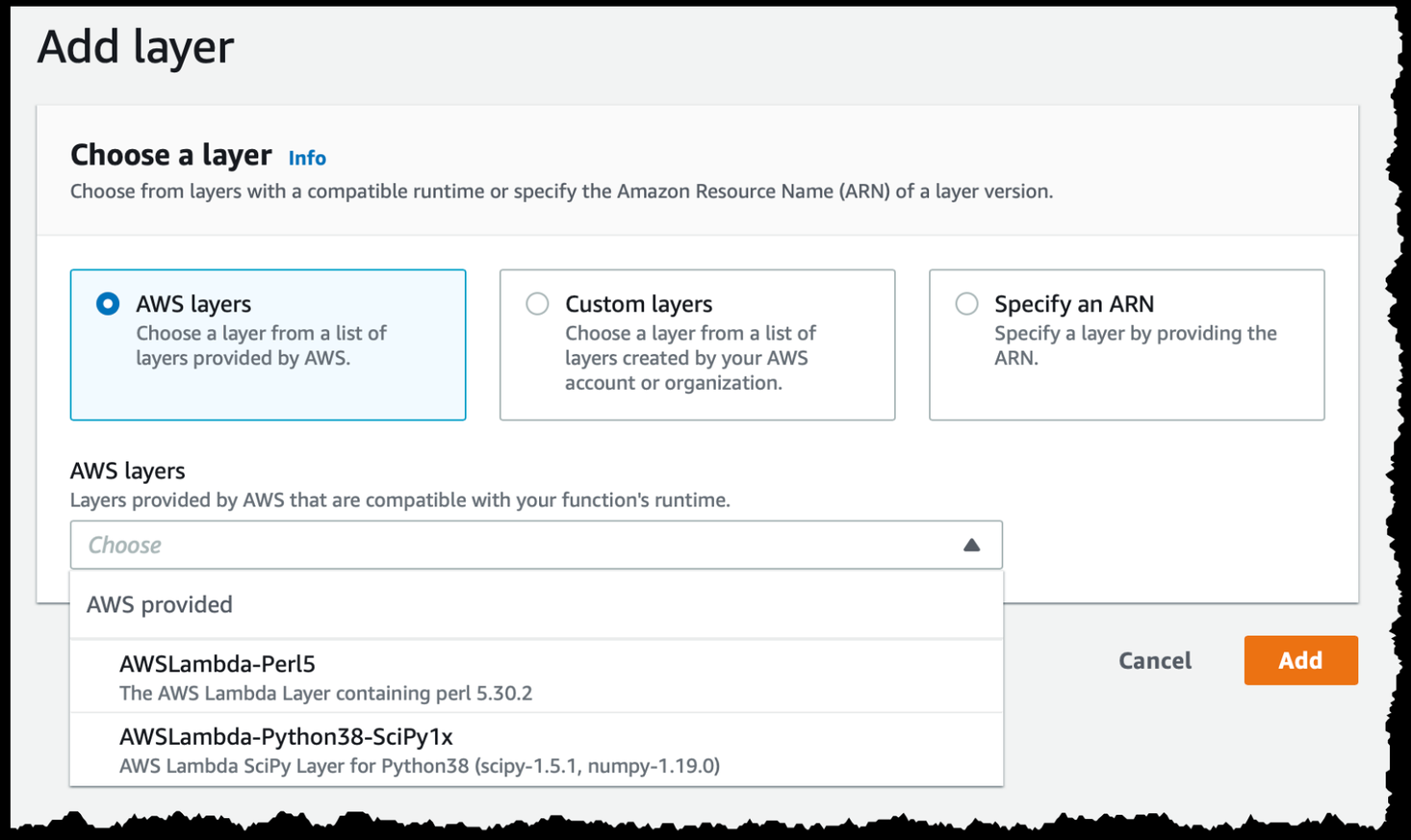
To run inference, you can combine all or a subset of the results



AWS Layer for Python: NumPy and SciPy

For example, for Python 3.8

- NumPy 1.19.0
- SciPy 1.5.1



Using Amazon EFS for hosting dependencies

- Extends the reach of Lambda to new uses cases
- Machine learning training
  - CPU-based, 15 minutes time limit
  - Improving results with ensemble forecasts or ensemble learning
- Machine learning inference
  - Using CPUs
  - Evaluate your latency requirements (sync vs async invocations)
  - Use Provisioned Concurrency to avoid cold starts
- You can use an Amazon Elastic Compute Cloud (Amazon EC2) instance to install dependencies and copy them to an Amazon EFS file system
  - Some tools created by the AWS community are great! For example
    - https://github.com/lambci/cmda

Machine learning inference (with Amazon EFS)

- Using PyTorch



```
{"bird_class": "106.Horned_Puffin"}
```



```
{"bird_class": "053.Western_Grebe"}
```

https://aws.amazon.com/blogs/aws/new-a-shared-file-system-for-your-lambda-functions/

Machine learning inference (with Amazon EFS)

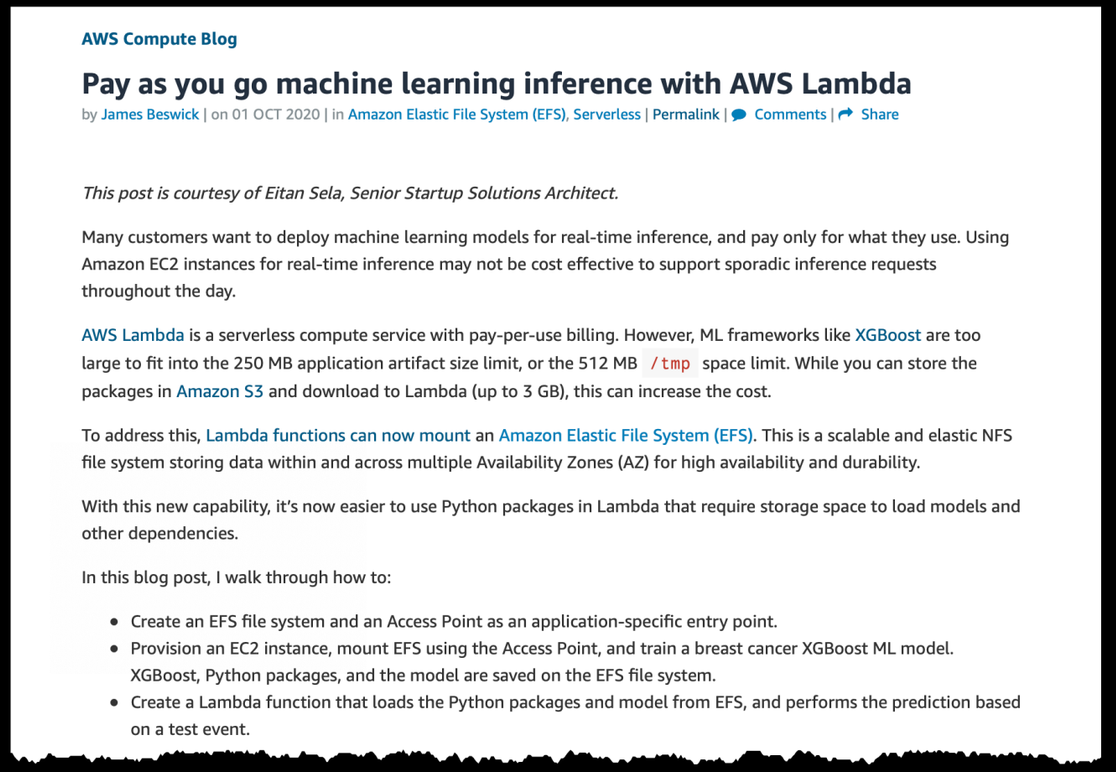
- Using TensorFlow



https://aws.amazon.com/blogs/compute/building-deep-learning-inference-with-aws-lambda-and-amazon-efs/

Machine learning inference (with Amazon EFS)

- Using XGBoost



AWS Compute Blog

Pay as you go machine learning inference with AWS Lambda

by James Beswick | on 01 OCT 2020 | in Amazon Elastic File System (EFS), Serverless | Permalink | Comments | Share

This post is courtesy of Eitan Sela, Senior Startup Solutions Architect.

Many customers want to deploy machine learning models for real-time inference, and pay only for what they use. Using Amazon EC2 instances for real-time inference may not be cost effective to support sporadic inference requests throughout the day.

AWS Lambda is a serverless compute service with pay-per-use billing. However, ML frameworks like XGBoost are too large to fit into the 250 MB application artifact size limit, or the 512 MB `/tmp` space limit. While you can store the packages in Amazon S3 and download to Lambda (up to 3 GB), this can increase the cost.

To address this, Lambda functions can now mount an Amazon Elastic File System (EFS). This is a scalable and elastic NFS file system storing data within and across multiple Availability Zones (AZ) for high availability and durability.

With this new capability, it's now easier to use Python packages in Lambda that require storage space to load models and other dependencies.

In this blog post, I walk through how to:

- Create an EFS file system and an Access Point as an application-specific entry point.
- Provision an EC2 instance, mount EFS using the Access Point, and train a breast cancer XGBoost ML model. XGBoost, Python packages, and the model are saved on the EFS file system.
- Create a Lambda function that loads the Python packages and model from EFS, and performs the prediction based on a test event.

https://aws.amazon.com/blogs/compute/pay-as-you-go-machine-learning-inference-with-aws-lambda/

Using data science dependencies in Python

- Add the following dependencies to your `requirements.txt`

```
numpy
scipy
scikit-learn
panda
matplotlib
ipython
jupyter
jupyterlab
papermill
```

Using data science dependencies in Python

- Mount Amazon EFS at instance launch, by default it will be on `/mnt/efs/fs1/`

- Then, you can use these commands (on Amazon Linux 2):

  - `sudo yum update –y # reboot if kernel is updated`

  - `sudo mkdir –p /mnt/efs/fs1/DataScience/lib`
  - `sudo chown -R ec2-user:ec2-user /mnt/efs/fs1/DataScience`

  - `sudo amazon-linux-extras install python3.8`

  - `pip3.8 install --user wheel`
  - `pip3.8 install -t /mnt/efs/fs1/DataScience/lib -r requirements.txt`

Using data science dependencies in Python

- Create Amazon EFS access point for `/DataScience`
  - On Amazon Linux, you can use `ec2-user`'s UID and GID
- Connect Lambda function to the Amazon Virtual Private Cloud (Amazon VPC)
  - To connect to other AWS services, such as Amazon S3 or Amazon DynamoDB, use Amazon VPC Endpoints
  - To reach the public internet, use private subnets + NAT Gateway
- Add file system to function using Amazon EFS access point
  - For example, mount under `/mnt/DataScience`
- Configure function environment to use dependencies
  - For example in Python `set PYTHONPATH = /mnt/DataScience/lib`

Random Forest Classifier

USING SCIKIT-LEARN FOR CLASSIFICATION

```
from sklearn.ensemble import RandomForestClassifier

import json

def lambda_handler(event, context):
    clf = RandomForestClassifier(random_state=0)
    X = [[ 1,  2,  3], [11, 12, 13]] # 2 samples, 3 features
    y = [0, 1] # classes of each sample
    clf.fit(X, y) # fitting the classifier
    A = [[4, 5, 6], [14, 15, 16], [3, 2, 1], [17, 15, 13]]
    result = {
        'type': 'RandomForestClassifier',
        'predict({})'.format(X): '{}'.format(clf.predict(X)),
        'predict({})'.format(A): '{}'.format(clf.predict(A))
    }
    return {
        'statusCode': 200,
        'body': json.dumps(result)
    }
```



Takeaways



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Takeaways

- Use **event-driven** architectures for metadata extraction, enrichment, and indexing
- For more advanced use cases, manage **dependencies** using an Amazon EFS file system to use tools like Scikit-learn and Matplotlib
- Simplify machine learning **inference** with **frameworks** such as PyTorch and TensorFlow running in Lambda functions
- Automate Jupyter **notebooks** execution using Papermill

Thank you!

Source code for all demos

 https://github.com/danilop/analyzing-data-aws-lambda

#StayInTouch

 @danilop

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.