mlmax

Release 0.2.0

Amazon Web Services

Jul 19, 2023

CONTENTS:

CHAPTER

ONE

ML MAX

ML Max is a set of example templates to accelerate the delivery of custom ML solutions to production so you can get started quickly without having to make too many design choices.

1.1 Quick Start

1. **ML Training Pipeline**: This is the process to set up standard training pipelines for machine learning models enabling both immediate experimentation, as well as tracking and retraining models over time.
2. **ML Inference Pipeline**: Deploys a model to be used by the business in production. Currently this is coupled quite closely to the ML training pipeline as there is a lot of overlap.
3. **Development environment**: This module manages the provisioning of resources and manages networking and security, providing the environment for data scientists and engineers to develop solutions.
4. **Data Management and ETL**: This module determines how the machine learning operations interacts with the data stores, both to ingest data for processing, managing feature stores, and for processing and use of output data. A common pattern is to take an extract, or mirror, of the data into S3 on a project basis.
5. **CICD Pipeline**: This module provides the guidance to setting up a continuous integration (CI) and continuous deployment (CD) pipeline, and automate the delivery of the ML pipelines (e.g., training and inference pipelines) to production using multiple AWS accounts (i.e., devops account, staging account, and production account.).

1.2 Help and Support

CHAPTER

TWO

ML TRAINING AND INFERENCE PIPELINE

The Training module manages the process to set up standard training pipelines for machine learning models enabling both immediate experimentation, as well as tracking and retraining models over time.The Inference module deploys a model to be used by the business in production. Often this is coupled quite closely to the ML training pipeline as there is a lot of overlap.

2.1 Design Principles

2.2 Quick Start

1. Clone repo

```
conda create --name <name> python=3.7
conda activate <name>
git clone https://github.com/awslabs/mlmax.git
```

2. Install dependencies

```
cd mlmax/modules/pipeline
pip install -r requirements.txt
```

3. Create the CloudFormation

```
cd modules/pipeline
python training_pipeline_create.py
python inference_pipeline_create.py
```

4. Deploy the CloudFormation

You will need an S3 bucket for this step.

```
# ./deploy.sh <target_env> <s3_bucket>
# ./deploy.sh dev sagemaker-us-east-1-1234
```

5. Run the Training Pipeline

```
# python training_pipeline_run -e <target_env>
python training_pipeline_run.py dev
```

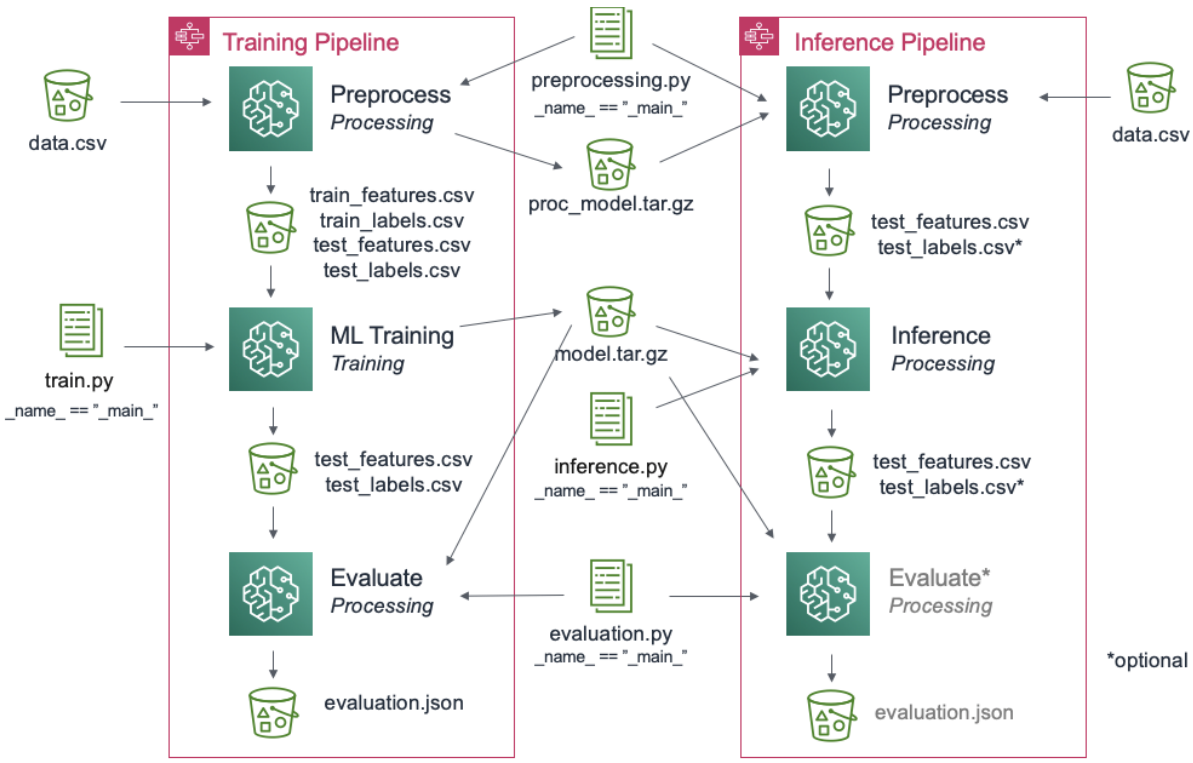
6. Run the Inference Pipeline

```
# python inference_pipeline_run -e <target_env>
python inference_pipeline_run.py dev
```

2.3 Architecture

The ML training and inference pipeline consists of Amazon Sakemaker running within Step Functions. The code included in this module:

- Generates the Cloudformation for the training and inference pipeline using the Data Science Step Functions SDK.
- Packages and deploys the Cloudformation, (creating the State Machines).
- Runs the traning and inference pipeline (executing the State Machines).



CHAPTER

THREE

DEVELOPMENT ENVIRONMENT

The Environment module manages the provisioning of resources and manages access controls, providing the environment for data scientists and engineers to develop solutions.

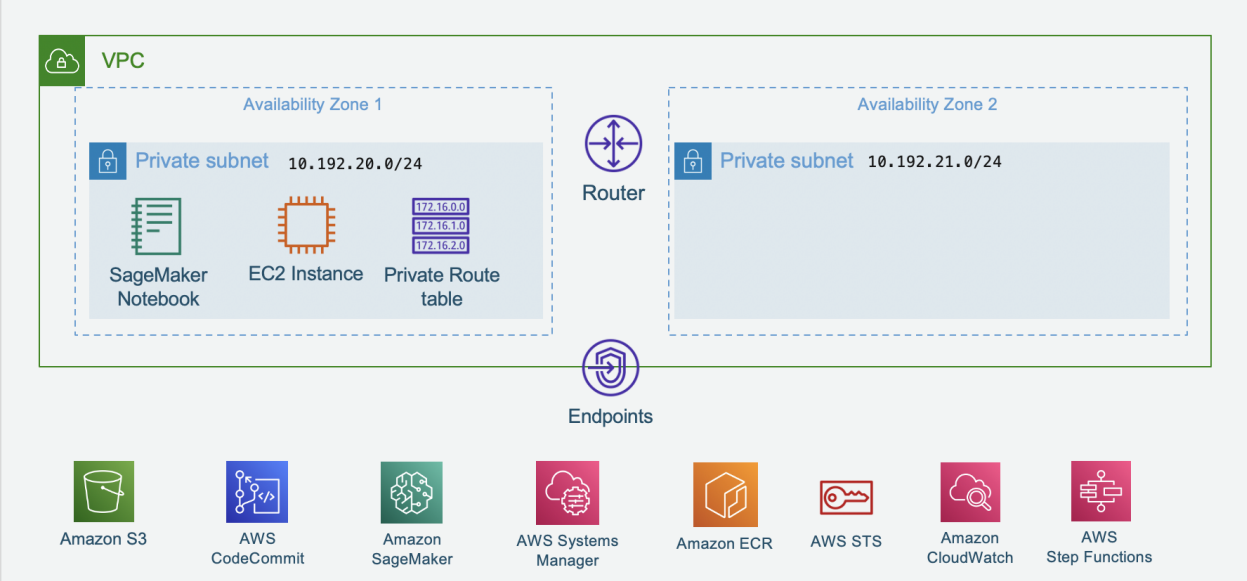
3.1 Design Principles

3.2 Quick Start

1. Update the following config in `config/config.ini`
   - KeyName: Existing EC2 key pair name that you have access to the private file
   - S3BucketName: Unique S3 bucket name for project
   - VpcCIDR: The Cidr range for the VPC
2. Prepare a S3 bucket in the same region to store cloudformation intermediate metedata. This could be an existing bucket or a new bucket. You must be able to write to this bucket. This is a different bucket than the one specified in Step 1.
3. To deploy, run the command `deploy.sh [stack-name] [cloudformation-bucket] [region]`
   - Cloundformation bucket be in the same region specified by `region` argument.
   - If no `region` argument is provided, default region in `.aws/config` will be used.
   - E.g. `deploy.sh my-stack my-cfn-s3bucket ap-southeast-1`

3.3 Architecture

Very often in a regulated industry such as Financial Services or Healthcare where data security is critical, the customer will have stringent requirements to be compliant for data science work. This template will setup the minimum service such as bastianless EC2 instance, SageMaker Notebook and S3 bucket for Data Scientist to start working on customer engagement.



S3

- Enforce service side encryption with customer managed key
- Data Scientist or developer is responsible to specify kmsid for AWSCLI, SDK or BOTO3 for any data upload to S3

SageMaker

- Encrypted EBS volume with customer manged key
- Restricted access to default encrypted S3 bucket

VPC Endpoint

The following endpoints have been added by default, additional endpoint can be added as necessary.

- s3
- git-codecommit, codecommit
- sagemaker.api, sagemaker.runtime
- ecr.api, ec.dkr
- sts
- logs
- ssm
- states

KMS

- Generate a customer managed key

VPC

- Two Private Subnets only across different avaiability zones

EC2

- SSM Agent to support remote SSH using exisitng key pair

- First verify that Session Manager Plugin is installed on your local workstation by running the command below or follow the instruction here to install SSM agent
- Your current active aws profile keypair should be the same as the deployment profile.

```
aws ssm start-session --target <ec2-instance-id>
```

- Update SSH config to be able to SSH to EC2 using command `ssh ec2-ssm`.

```
# Add the following in your SSH config

Host ec2-ssm
    HostName <ec2-instance-id>
    User ec2-user
    IdentityFile /path/to/keypair/pemfile
    ProxyCommand sh -c "aws ssm start-session --target %h --document-name AWS-
↪StartSSHSession --parameters 'portNumber=%p'"
```

3.4 Security Patching

It is recommended that you patch the EC2 instance regularly whenever there is security updates. Run the following commands in EC2 for patching.

```
sudo yum-config-manager --disable libnvidia-container
sudo yum-config-manager --disable neuron
sudo yum-config-manager --disable nvidia-container-runtime
sudo yum-config-manager --disable nvidia-docker
sudo yum update-minimal --sec-severity=critical,important --bugfix
```

CHAPTER

FOUR

DATA MANAGEMENT AND ETL

This module determines how the machine learning operations interact with the data stores to ingest data for processing, manage feature stores, and for processing and use of output data. A common pattern is to take an extract, or mirror of the data into S3 on a per project basis.

4.1 Quick Start

Prerequisites:

The data used in this example is taken from the NYC TLC Trip Record Data, which was downloaded using the tools provided by https://github.com/toddwschneider/nyc-taxi-data. These tools can be used to download the entire dataset from scratch, or to update an existing dataset.

You will need to upload the dataset to an S3 bucket. The folder structure should look something like this:

```
$ aws s3 ls s3://S3InputBucket/S3InputPrefix/ --human-readable
                           PRE unaltered/
2020-10-13 13:47:21  364.0 KiB central_park_weather.csv
2020-10-13 13:47:21   45.7 KiB fhv_bases.csv
2020-10-13 13:47:21   81.8 MiB fhv_tripdata_2015-01.csv
2020-10-13 13:47:21   93.3 MiB fhv_tripdata_2015-02.csv
...
2020-10-13 13:51:20    1.2 GiB fhvhv_tripdata_2019-02.csv
2020-10-13 13:51:20    1.4 GiB fhvhv_tripdata_2019-03.csv
...
2020-10-13 13:53:16    1.1 MiB green_tripdata_2013-08.csv
2020-10-13 13:53:17    7.3 MiB green_tripdata_2013-09.csv
...
2020-10-13 13:54:32    2.4 GiB yellow_tripdata_2009-01.csv
2020-10-13 13:54:32    2.2 GiB yellow_tripdata_2009-02.csv
...
2020-10-13 14:21:57   30.2 MiB yellow_tripdata_2020-05.csv
2020-10-13 14:21:58   47.9 MiB yellow_tripdata_2020-06.csv
```

1. Clone repo

```
conda create -y -n <name> python=3.7
conda activate <name>
git clone https://github.com/awslabs/mlmax.git
```

2. Install dependencies

```
cd mlmax/modules/data
pip install -r requirements.txt
```

3. Create the CloudFormation

```
python data_pipeline_create.py
```

4. Deploy the CloudFormation

You will need an S3 bucket for this step. After the deployment, the pipeline will be scheduled to run daily.

```
# ./deploy.sh <target_env> <s3_bucket>
# ./deploy.sh dev sagemaker-us-east-1-1234
```

5. Manually Run the Data Pipeline

```
# python data_pipeline_run -e <target_env>
python data_pipeline_run.py dev
```

CHAPTER

FIVE

UTILITIES

5.1 Screening

See here for how to check whether access has been granted for various SageMaker API calls.

5.2 Tips for Notebooks

See here for how to configure your Jupyter Notebook to accept local imports.

5.3 Vim, Tmux and Zsh

See here for how to configure your EC2 instance with vim, tmux, and zsh.

CHAPTER

SIX

PRFAQ

Now ML Engineers and Data Scientists can quickly create and use production-ready ML solutions on AWS. mlmax provides example templates for the delivery of custom ML solutions to production so you can get started quickly without having to make too many design choices.

Q: What is the motivation for mlmax?

Delivering ML solutions to production is hard. It is difficult to know where to start, what tools to use, and whether you are doing it right. Often each individual professional does it a different way based on their individual experience or they use prescribed tools developed within their company. Either way this requires a lot of investment of time to firstly decide what to do and secondly to implement and maintain the infrastructure. There are many existing tools that make parts of the process faster but many months of work is still required to tie these together to deliver robust production infrastructure.

Q: What is mlmax?

ML Max is a set of example templates for the delivery of custom ML solutions to production so customers can get started quickly without having to make too many design choices.

Q: What modules are currently implemented?

1. **ML Training Pipeline**: This is the process to set up standard training pipelines for machine learning models enabling both immediate experimentation, as well as tracking and retraining models over time.
2. **ML Inference Pipeline**: Deploys a model to be used by the business in production. Currently this is coupled quite closely to the ML training pipeline as there is a lot of overlap.
3. **Development environment**: This module manages the provisioning of resources and manages networking and security, providing a the environment for data scientists and engineers to develop solutions.

Q: What is planned for the future?

Future work will include additional features such as scheduling, metadata management, and monitoring.

Q: How can I use training and inference pipeline?

There are a couple basic steps to go through to set up your training and inference pipeline:

- **Define the workflow**: in this step you are defining the input and output placeholders, which will later be supplied during runtime.
- **Create the workflow**: in this step you will create the underlying infrastructure including the Step Machine State Machine
- **Define the inputs/outputs**: in this step you put the raw data onto S3, both for training and for inference.
- **Define the runtime scripts**: in this step you define the logic for preprocessing, evaluation, training and inference. These are python scripts that can be run locally as well as within SageMaker.

- **Run the pipeline**: in this step you will create a Step Machine execution. It will upload the python files to S3 where they will be retrieved by the Step Functions. All meta-data will be saved to S3 including the scripts, metrics, model, and preprocessing model.

See modules/pipeline for quick start documentation.

Q: What are the design principles that the Training and Inference pipeline was created with?

- **Separate definition and runtime for training and inference pipelines.** This enable effective collaboration between the ML Ops Engineer and the Data Scientist. The ML Engineer can focus on managing and versioning the underlying infrastructure and the Data Scientist can focus on the development of the ML model, the input data, and logic for the data, preprocessing and evaluation.
- **Ensure the same code for pre-processing can be used for training and inference.** The exact some logic for data transformation needs to be used for training and inference time so that the consistency of results is achieved without silent errors.
- **Enforce traceability between components of the training and inference pipelines.** Training job should point to which code was used for pre-processing, which set of data was used to source it, etc…
- **Code should provide reproducible results wherever it is run.** Code should be able to be run locally, within EC2, on SageMaker, etc… This enables easy development, and debugging.
- **Promote modularity in the development of Training and Inference solutions.** There are separate steps for the each of the core components of the pipeline including preprocessing, training, inference, and evaluation. Where there is overlap, components are reused.

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search