Hewlett Packard Enterprise

Fashion MNIST Kubeflow Pipeline with GPU Runbook

Date Prepared: Feb 2021

Hewlett Packard Enterprise

Document Information

Project Name	**HPE**		
Project Owner		Document Version No	0.1
Quality Review Method			
Prepared By	vivek-singh.bhadauriya@hpe.com	Preparation Date	30-Apr-2021
Reviewed By	Jai Kakkar	Review Date	

Table of Contents

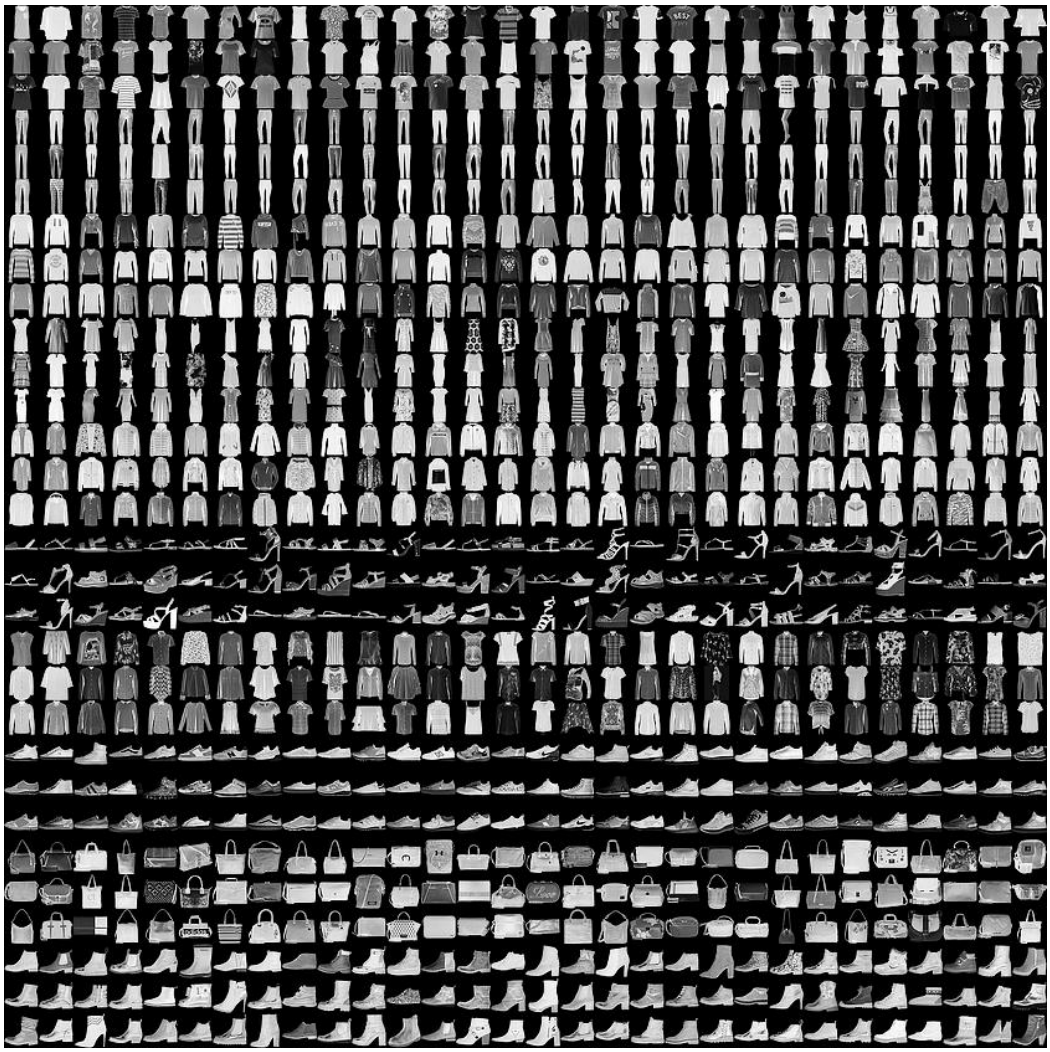
1 SUMMARY

This runbook will demonstrate set up Kubeflow Notebook on HPECP 5.3 and build a Kubeflow Pipeline which will trained model using GPU.

Use Case

Fashion-MNIST is a dataset of Zalando's article images consisting of a training set of 60,000 examples and a test set of 10,000 examples. Each example is a 28x28 grayscale image, associated with a label from 10 classes.

The images show individual articles of clothing at low resolution (28 by 28 pixels), as seen here:



Steps

- Create PVC.
- Upload required file to PVC.
- Submit the pipeline using Kubeflow Notebook.

Source Code: https://github.hpe.com/hpe/field-resources/tree/master/HPECP-5.3/Kubeflow/Fashion-MNIST-Keras-GPU-Pipeline

2 PREPARE THE DATASET

In this step we are going to create PVC for the use case.

Log in to Kubernetes Master Host and create a file name **fashion-mnist-pvc.yaml**.

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: fashion-mnist-pvc
spec:
  accessModes:
  - ReadWriteMany
  resources:
    requests:
      storage: 2Gi
```

Command to apply the **fashion-mnist-pvc.yaml**.

```
kubectl create -f fashion-mnist-pvc.yaml -n kubeflow
```

Download required files from below links:

- https://raw.githubusercontent.com/zalandoresearch/fashion-mnist/master/data/fashion/t10k-images-idx3-ubyte.gz
- https://raw.githubusercontent.com/zalandoresearch/fashion-mnist/master/data/fashion/t10k-labels-idx1-ubyte.gz
- https://raw.githubusercontent.com/zalandoresearch/fashion-mnist/master/data/fashion/train-images-idx3-ubyte.gz
- https://raw.githubusercontent.com/zalandoresearch/fashion-mnist/master/data/fashion/train-labels-idx1-ubyte.gz

After downloading the above files, we need to deploy a pod with attaching the PVC created to copy files to volume mount.

Create a **pod.yaml** file.

```
apiVersion: v1
kind: Pod
metadata:
    name: dataaccess
spec:
    containers:
    - name: alpine
      image: alpine:latest
      command: ['sleep', 'infinity']
      volumeMounts:
      - name: mypvc
        mountPath: /data
    volumes:
    - name: mypvc
      persistentVolumeClaim:
        claimName: fashion-mnist-pvc
```

Command to apply the **pod.yaml**.

```
kubectl create -f pod.yaml -n kubeflow
```

Once the pod with name **dataaccess** come to ready state execute the below command:

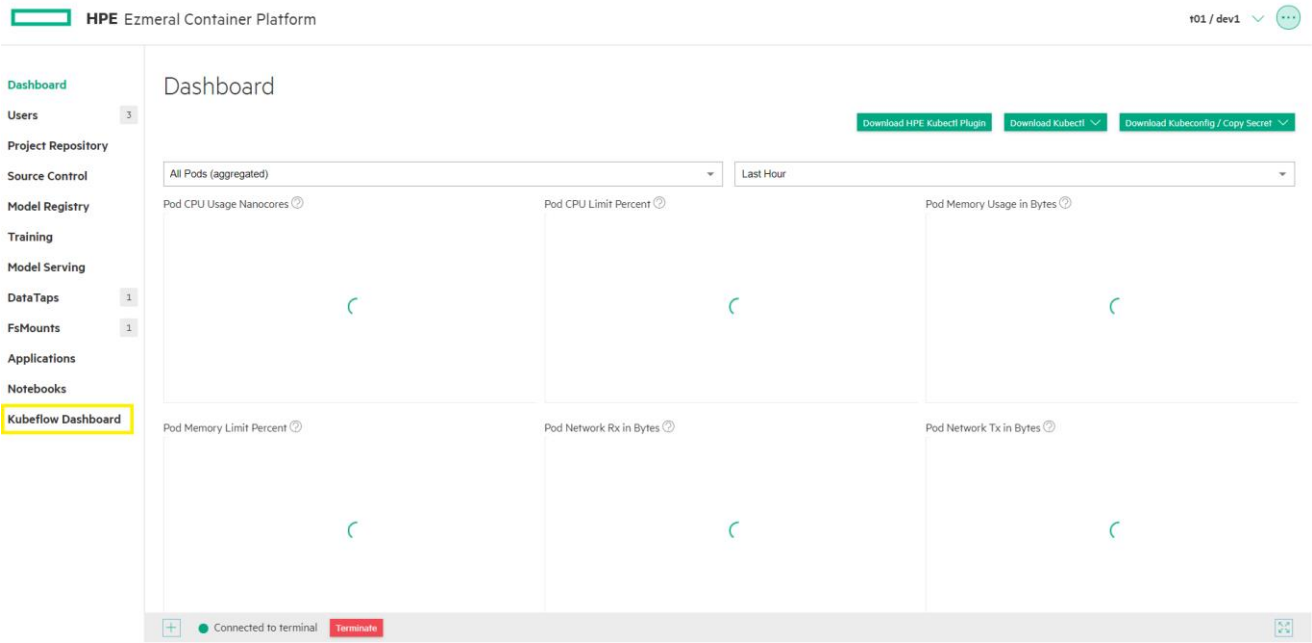
```
kubectl cp t10k-images-idx3-ubyte.gz dataaccess:/data -n kubeflow
kubectl cp t10k-labels-idx1-ubyte.gz dataaccess:/data -n kubeflow
kubectl cp train-images-idx3-ubyte.gz dataaccess:/data -n kubeflow
kubectl cp train-labels-idx1-ubyte.gz dataaccess:/data -n kubeflow
```

After copying the file will delete the pod.

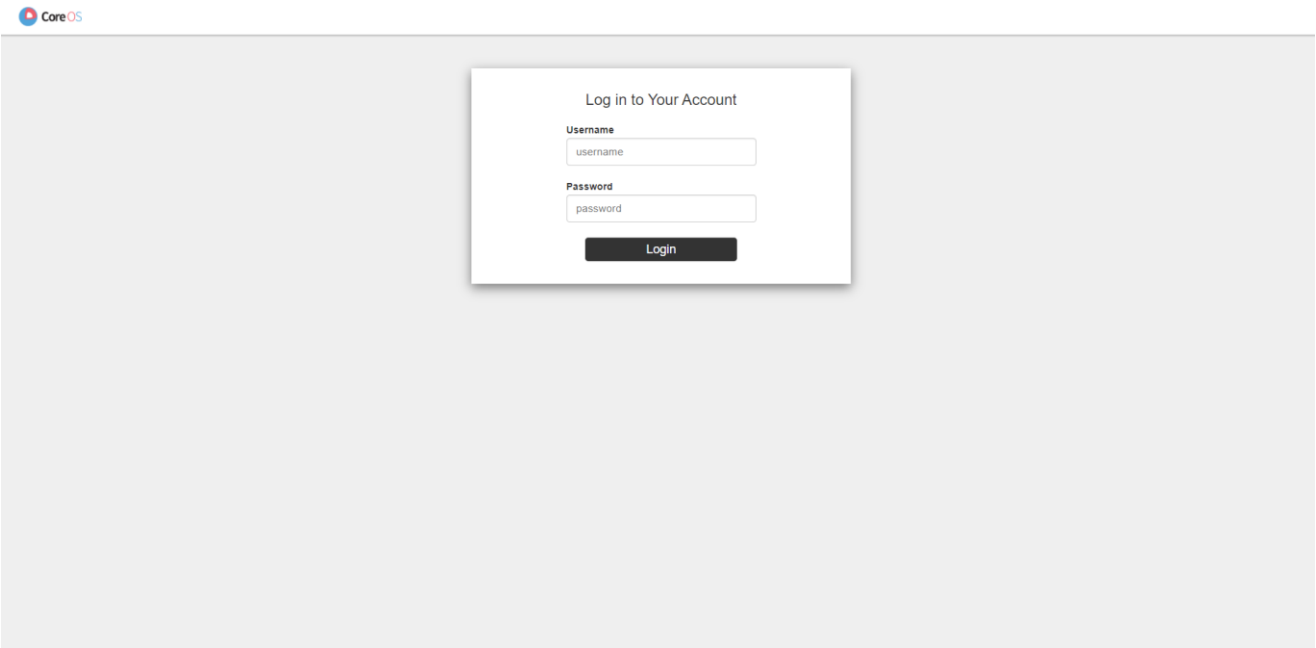
```
kubectl delete -f pod.yaml -n kubeflow
```

3 DEPLOY KUBEFLOW NOTEBOOK

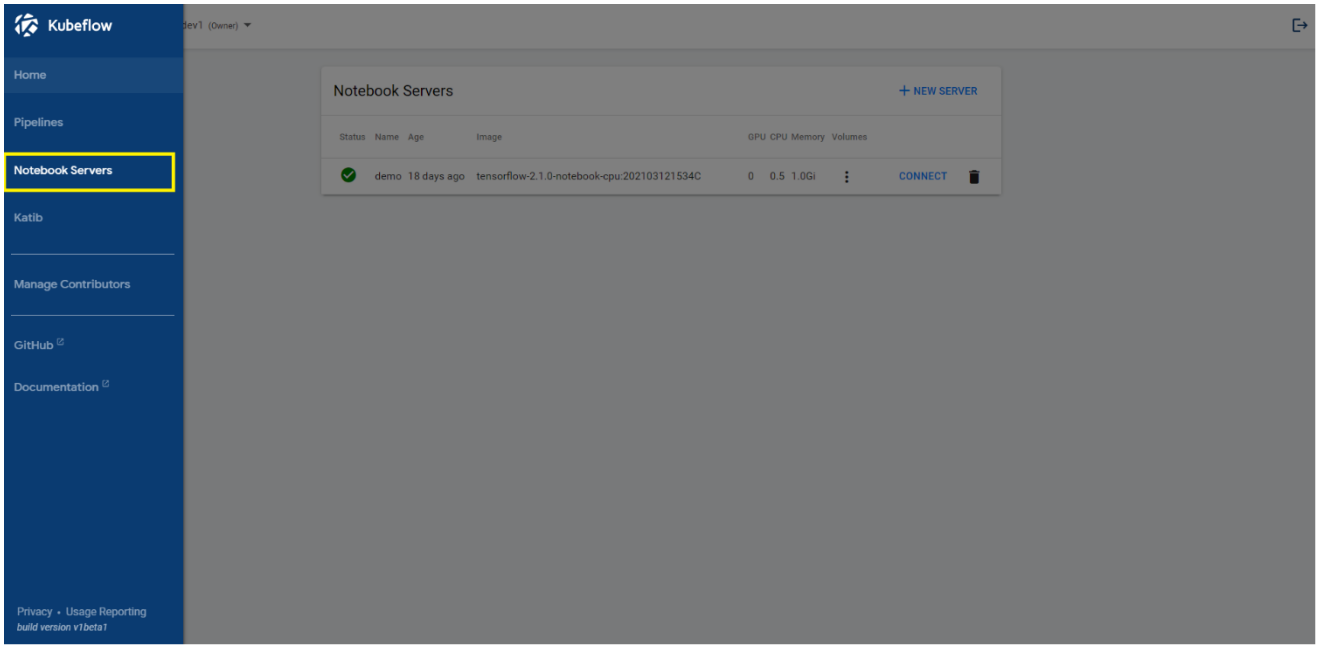
Go to Tenant and click on Kubeflow Dashboard.



Login to Kubeflow UI with AD user.



After Login on Kubeflow UI, click on Notebook Server.



If there is no Notebook Server, you can create new one. Click on Connect, that will redirect to Notebook Server and the upload the **Fashion-MNIST-GPU-KF-Pipeline.ipynb** notebook**.**

4 EXECUTE THE NOTEBOOK

After Uploading the notebook, will execute the notebook cell to submit the pipeline.

The main importing to place our pipeline step container to spin up on GPU host is the below function.

https://kubeflow-pipelines.readthedocs.io/en/latest/source/kfp.dsl.html?highlight=GPU#kfp.dsl.UserContainer.set_gpu_limit

```
# 2. Create MNIST training component.
mnist_training_container = train_op(data_path, model_file) \
                              .apply(onprem.mount_pvc(PVC_NAME, 'local-storage', DATA_PATH)).set_gpu_limit(1)

# 3. Create MNIST prediction component.
mnist_predict_container = predict_op(data_path,
                                     model_file,
                                     image_number
                                     ) \
                              .apply(onprem.mount_pvc(PVC_NAME, 'local-storage', DATA_PATH)).set_gpu_limit(1)

mnist_predict_container.after(mnist_training_container)
```

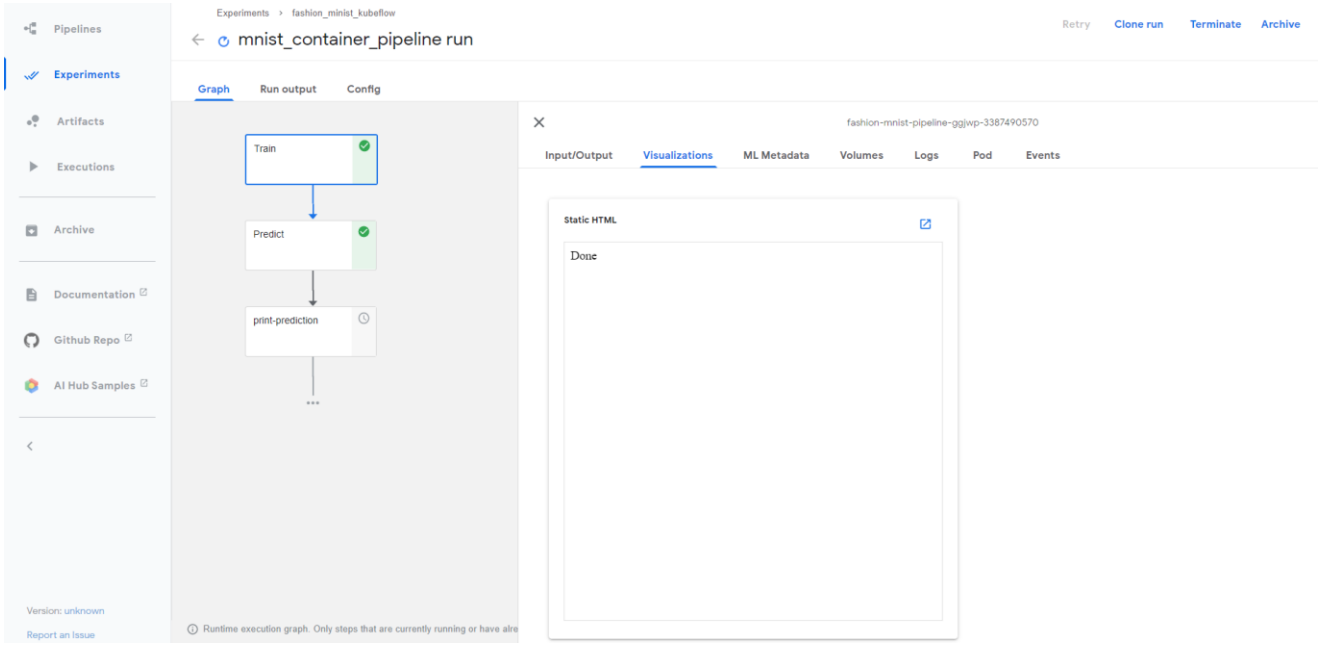
Once the pipeline is submitted.

```
Experiment details.
Run details.
Out[119]: RunPipelineResult(run_id=930f41f7-d161-4e55-b73e-5102fa48e220)
```

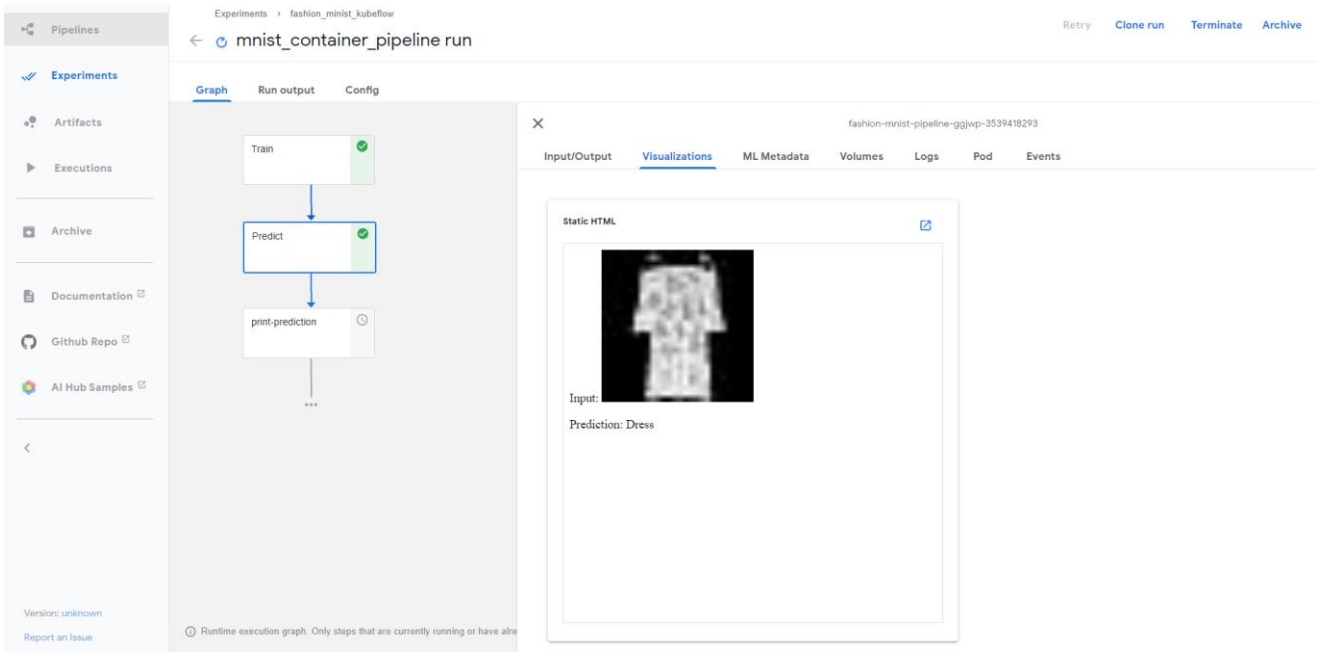
Click on **Run details**.

It will redirect you to the Kubeflow UI and display the Pipeline execution.

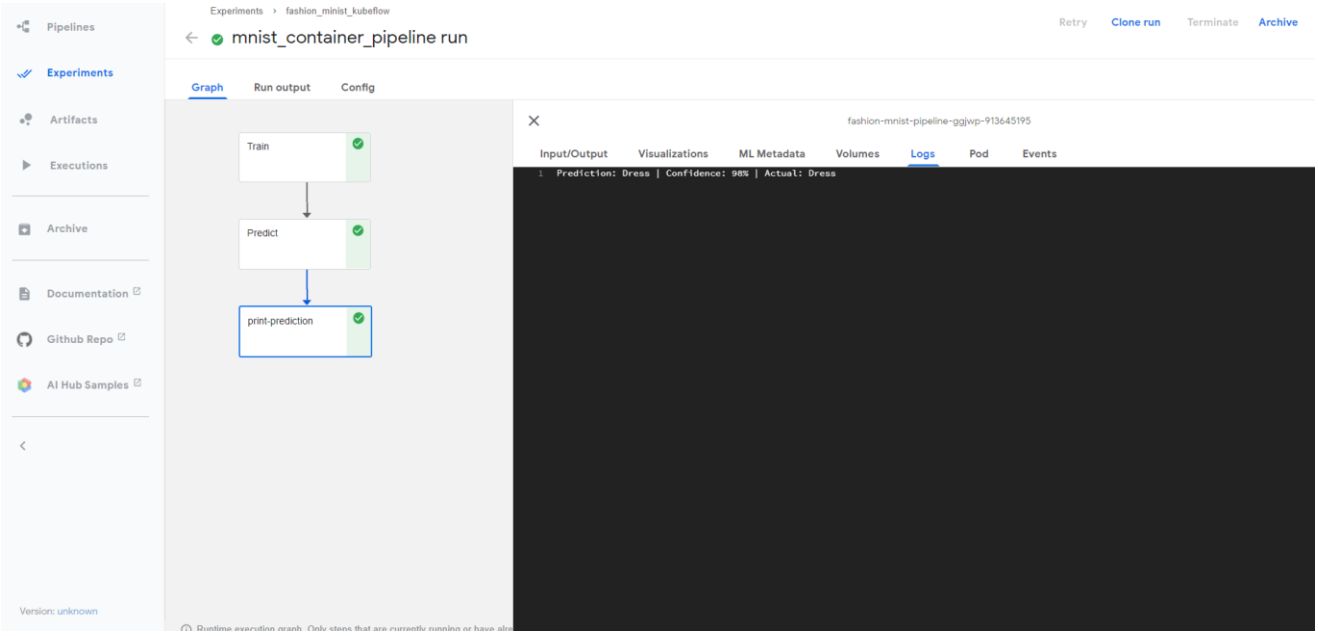
Training:



Prediction:



Prediction Result:



Model Metrics: