Hewlett Packard Enterprise

BUILDING DYNAMIC MACHINE LEARNING PIPELINES WITH KUBEDIRECTOR

Tom Phelan, Fellow Software Organization @Hewlett Packard Enterprise
Kartik Mathur, Master Technologist @Hewlett Packard Enterprise
Donald Wake, Technical Marketing Engineer @Hewlett Packard Enterprise

October
2020

DEPLOYING ENTERPRISE AI AND ML AT SCALE IS CHALLENGING

- Complex setup and execution
- Models become outdated due to fast changing and growing data sets
- Kubernetes offers enormous capabilities but can be challenging to maintain

"Over 60% of models developed with the intention of operationalizing them were never actually operationalized. There are many reasons for this, but a crucial one is a lack of tools to enable and facilitate operationalization, which is not just about deployment."
- Gartner

AGENDA

- Overview
- Stateless & Stateful Applications
- KubeDirector
- Machine Learning Pipelines
- Launching KDClusters
- Training The Model
- Registering The Model
- Creating Inference Deployment
- Serving Queries

STATELESS APPLICATIONS

- **Stateless**
  - Each application service instance is configured identically
  - All information stored remotely
  - “Remotely” refers to some persistent storage that has a life span different from that of the container
  - Frequently referred to as “cattle”

STATEFUL APPLICATIONS

- **Stateful**
  - Each application service instance is configured differently
  - Critical information stored locally
  - “Locally” means that the application running in the container accesses the information via file system reads/writes rather than some remote access protocol
  - Frequently referred to as “pets”

KUBERNETES - COMPONENTS

- Objects
- Pods
- Statefulsets
- PersistentVolumes
- Operators
- Custom Resource Definitions

KUBEDIRECTOR

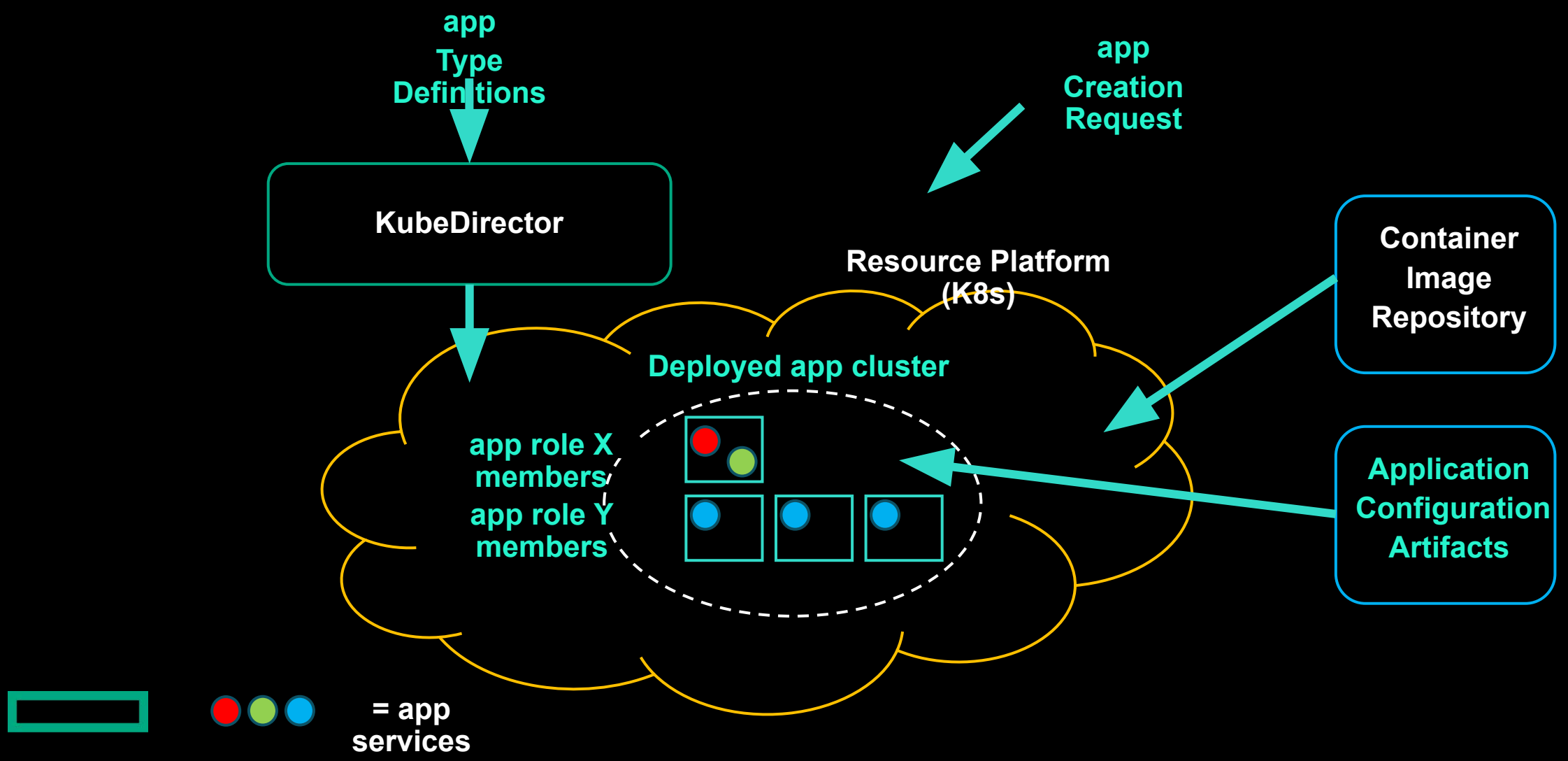
- KubeDirector is a K8s “custom controller”
- Introduced to the open source community to address stateful application deployment in standard Kubernetes clusters
- In the latest release (version 0.5.1), KubeDirector now allows multiple clusters to share data very easily using a new feature called ***Connections***.

HOW KD SUPPORTS LEGACY STATEFUL APPS

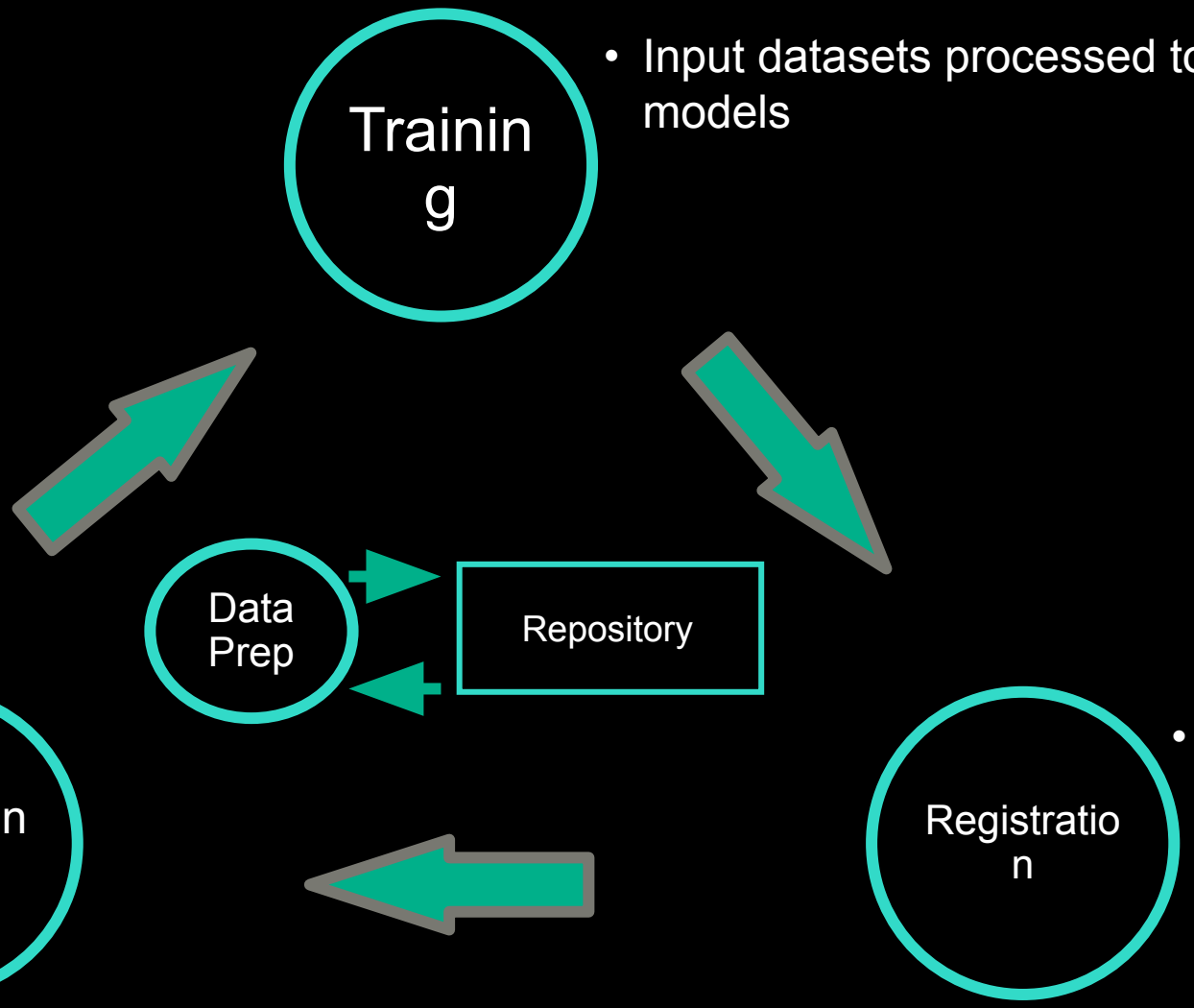
- Appconfig orchestration layer for every pod
- Guestconfig hooks, --configure, --addnodes, --delnodes
- Config metadata – cluster view
- Configcli to query configmeta

DEPLOY KUBEDIRECTOR TO K8S

```
kubectl create -f kubedirector/deployment.yaml
```



BASIC MACHINE LEARNING PIPELINE



- Input datasets processed to create ML models
- Models are identified & characterized
- Selected models are made available for answering queries

KUBEDIRECTOR APPLICATIONS (KDAPPS) FOR OUR ML PIPELINE

- A kdapp is a custom resource (CR)
- The kdapp instructs KubeDirector on how a particular kind of virtual application cluster should be deployed and managed
- These example kdapps are available online

KDAPPS

training-engine: A training deployment kdapp

jupyter-notebook: A Jupyter Notebook kdapp

deployment-engine: An inferencing deployment kdapp.

https://github.com/bluek8s/kubedirector/tree/master/deploy/example_cat

KDAPP EXAMPLE

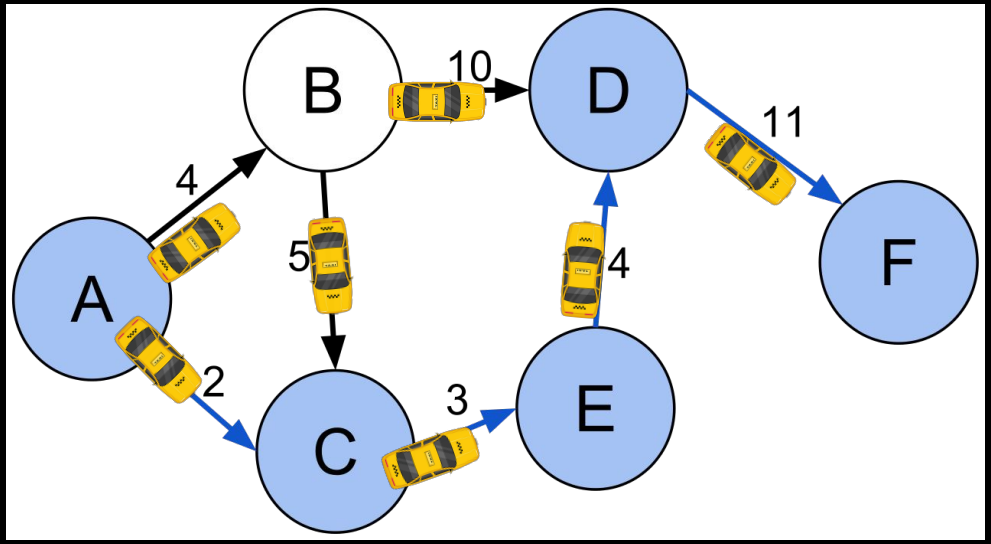

```
{
    "apiVersion": "kubedirector.hpe.com/v1beta1",
    "kind": "KubeDirectorApp",
    "metadata": {
        "name": "training-engine",
        "labels": {
            "kubedirector.hpe.com/AIML_category": "Training"
        }
    },

    "spec": {
        "categories": [
            "AIML/Training"
        ],
        "config": {
            "configMeta": {
                "ml_engine": "python"
            },
            "roleServices": [{
                    "roleID": "RESTServer",
                    "serviceIDs": [
                        "gunicorn",
                        "ssh"
                    ]
                },
                {
                    "roleID": "LoadBalancer",
                    "serviceIDs": [
                        "gunicorn",
                        "haproxy-stats",
                        "haproxy-train"
                    ]
                },
                {
                    "roleID": "controller",
                    "serviceIDs": [
                        "ssh",
                        "jupyter-nb"
                    ]
                }
            ],
            "selectedRoles": [
                "RESTServer",
                "LoadBalancer",
                "controller"
            ]
        },
        "distroID": "hpecp/training-engine",
        "label": {
            "description": "Toolkit: TensorFlow, Scikit-Learn, PyTorch, Keras, XGBoost, LightGBM, hyperopt, Horovod; Core: Numpy,
Seaborn, Plotly, Bokeh",
            "name": "ML Training Toolkit, with GPU support",
        },
        "roles": [{
                "cardinality": "1",
                "configPackage": {
                    "packageURL": "file:///opt/configscript/flask_configure.tgz"
                },
                "id": "LoadBalancer",
                "imageRepoTag": "bluedata/kd-api-serving:1.0"
            },
            {
                "cardinality": "1+",
                "configPackage": {
                    "packageURL": "file:///opt/configscript/flask_configure.tgz"
                },
                "id": "RESTServer",
```

```
                "configPackage": {
                    "packageURL": "file:///opt/configscript/flask_configure.tgz"
                },
                "id": "RESTServer",
                "imageRepoTag": "bluedata/kd-api-serving:1.0"
            },
                "cardinality": "1",
                "configPackage": {
                    "packageURL": "file:///opt/configscript/appconfig.tgz"
                },
                "id": "controller",
                "imageRepoTag": "bluedata/kd-training:1.0"
            }
        ],
        "services": [{
                "endpoint": {
                    "isDashboard": true,
                    "port": 8081,
                    "urlScheme": "http"
                },
                "id": "haproxy-stats",
                "label": {
                    "name": "Model serving request balancer stats"
                }
            },
            {
                "endpoint": {
                    "isDashboard": false,
                    "port": 22
                },
                "id": "ssh",
                "label": {
                    "name": "SSH"
                }
            },
            {
                "endpoint": {
                    "hasAuthToken": true,
                    "isDashboard": false,
                    "port": 10001,
                    "urlScheme": "http"
                },
                "id": "gunicorn",
                "label": {
                    "name": "API Server"
                }
            },
            {
                "endpoint": {
                    "hasAuthToken": true,
                    "isDashboard": false,
                    "path": "/train",
                    "port": 32700,
                    "urlScheme": "http"
                },
                "exported_service": "AIML/Training",
                "id": "haproxy-train",
                "label": {
                    "name": "Training API Server"
                }
            },
            {
                "id": "httpd",
                "label": {
                    "name": "Apache HTTP Server"
                }
            },
            {
                "endpoint": {
                    "isDashboard": true,
                    "path": "/",
```

EXAMPLE MACHINE LEARNING PROBLEM: FIND TAXI RIDE TIMES

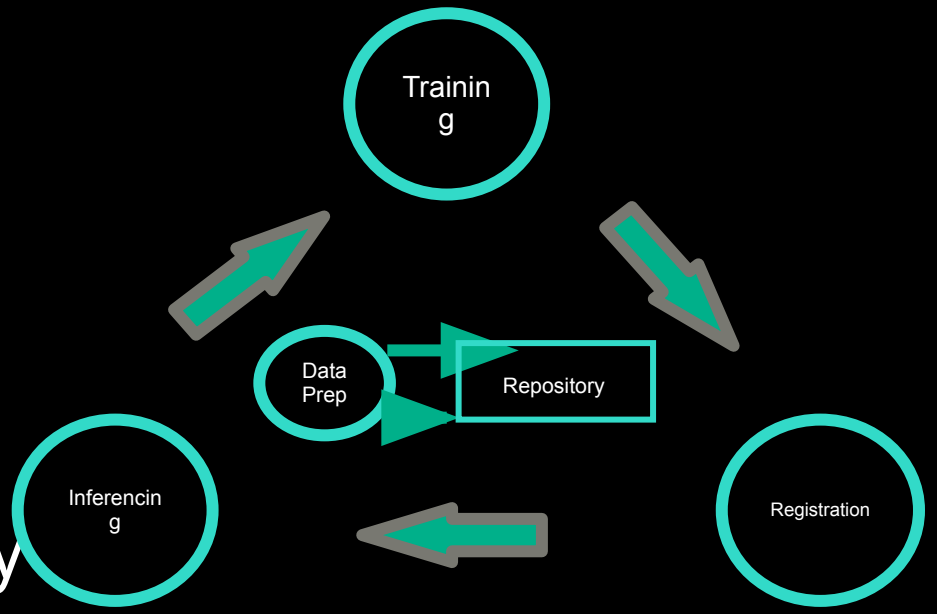
ML Example Problem Description

Create an Artificial Intelligence application that can predict the travel time for a proposed taxi ride. The application must dynamically adjust to new datasets as they become available.



REGISTER YOUR KDAPPS

Assuming that KubeDirector is already deployed and running in the Kubernetes cluster, these kdapp CRs can be created to register apps with KubeDirector, e.g. by using kubectl:




```
kubectl create -f cr-app-training-engine.json
kubectl create -f cr-app-jupyter-notebook.json
kubectl create -f cr-app-deployment-engine.json
```

KUBEDIRECTOR VIRTUAL CLUSTER - KDCLUSTER

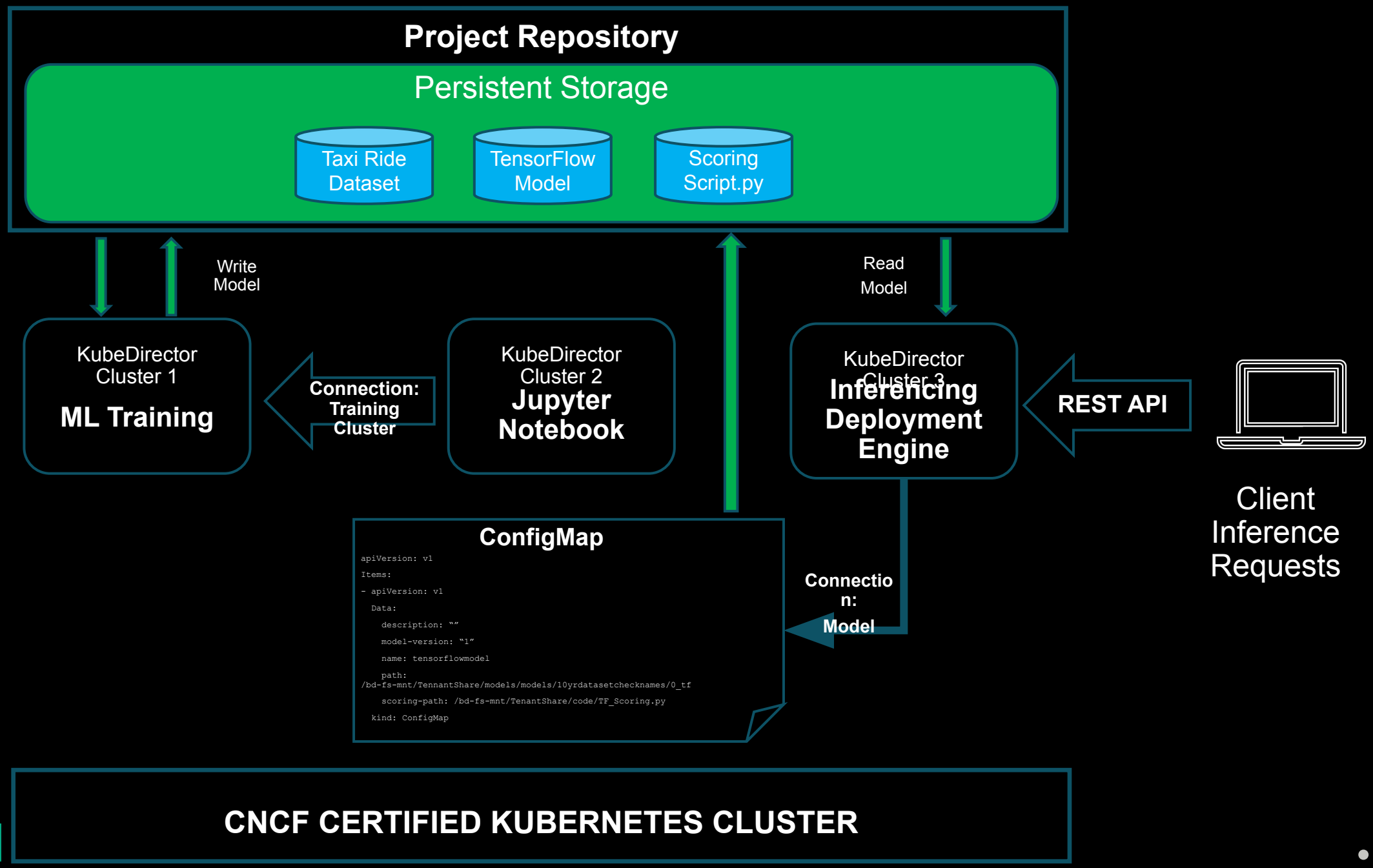
- Once a kdapp is created, an instance of that app can be deployed by creating a KubeDirector virtual cluster (kdcluster) CR
- A kdcluster identifies the desired kdapp and specifies runtime configuration parameters, such as the size and resource requirements of the virtual cluster

```yaml
apiVersion: "kubedirector.hpe.com/v1beta1"
kind: "KubeDirectorCluster"
metadata:
  name: "training-environment"
spec:
  app: training-engine
  roles:
  - id: RESTServer
    members: 1
    resources:
      limits:
        cpu: "1"
        memory: 2Gi
      requests:
        cpu: "1"
        memory: 2Gi
  - id: LoadBalancer
    members: 1
    resources:
      limits:
        cpu: "1"
        memory: 2Gi
      requests:
        cpu: "1"
        memory: 2Gi
  - id: controller
    members: 1
    resources:
      limits:
        cpu: "2"
        memory: 8Gi
      requests:
        cpu: "2"
        memory: 8Gi
```

KDAPP CONNECTIONS

- Attach additional resources to kdcluster
  - Connected endpoints
  - Secret/configmap/kdcluster/modelCR
- Maintain data-structure for cluster to consume the additional resources
- Dynamic update of advertised metadata
- Provides utility tools to query the data structure
- Event triggered hooks –reconnect for dynamic reconfiguration

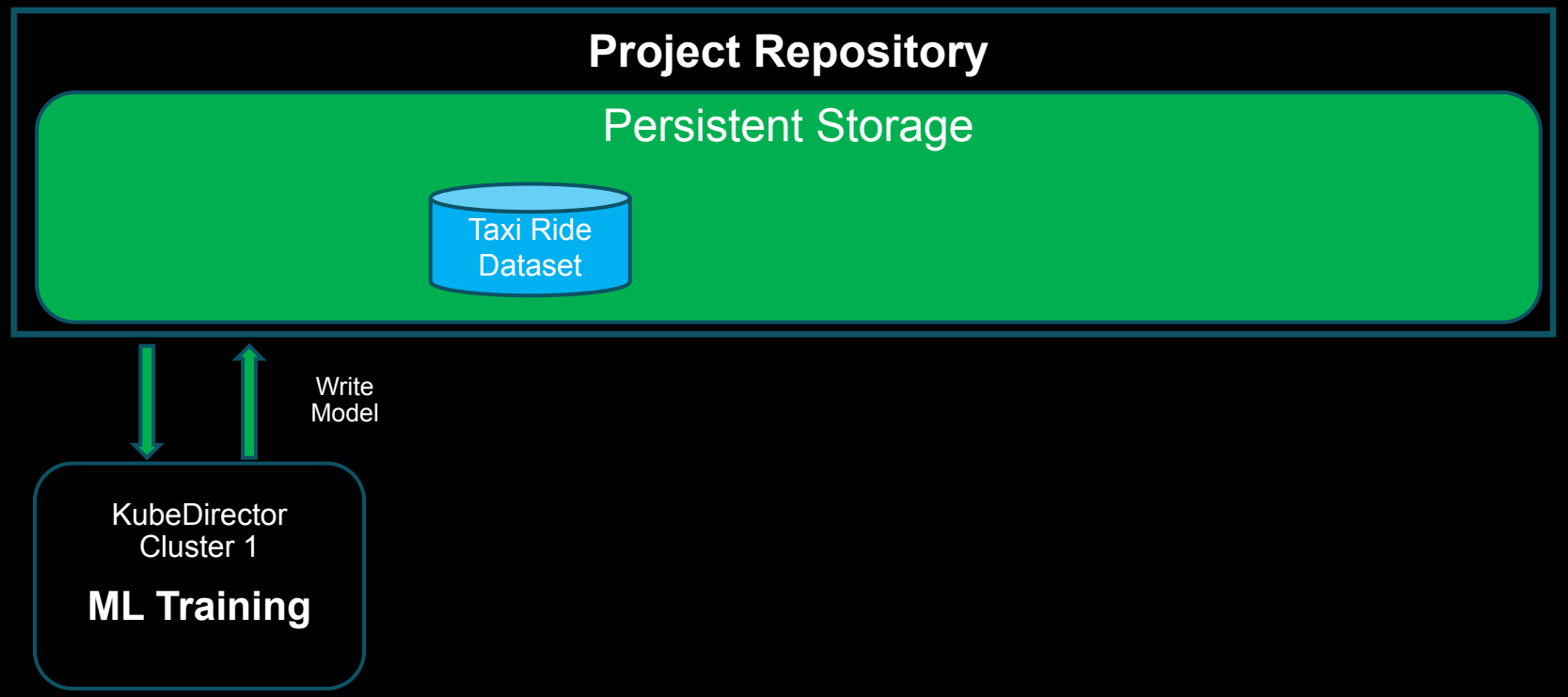
```yaml
apiVersion: "kubedirector.hpe.com/v1beta1"
kind: "KubeDirectorCluster"
metadata:
  name: "deployment-engine"

spec:
  app: model-serving
  connections:
    configmaps:
    - "convolutional-nn-model"
    - "fullyconnected-nn-model"
  roles:
  - id: RESTServer
    members: 1
    resources:
      requests:
        memory: "6Gi"
        cpu: "2"
      limits:
        memory: "6Gi"
        cpu: "2"
  - id: LoadBalancer
    members: 1
    resources:
      requests:
        memory: "2Gi"
        cpu: "1"
      limits:
        memory: "2Gi"
        cpu: "1"
```


Project Repository
Persistent Storage
Taxi Ride Dataset
TensorFlow Model
Scoring Script.py
Write Model
Read Model
KubeDirector Cluster 1
ML Training
Connection: Training Cluster
KubeDirector Cluster 2
Jupyter Notebook
KubeDirector Cluster 3
Inferencing Deployment Engine
REST API
Client Inference Requests
ConfigMap
apiVersion: v1
Items:
- apiVersion: v1
Data:
description: “”
model-version: “1”
name: tensorflowmodel
path: /bd-fs-mnt/TennantShare/models/models/10yrdatasetchecknames/0_tf
scoring-path: /bd-fs-mnt/TenantShare/code/TF_Scoring.py
kind: ConfigMap
Connection: Model
CNCF CERTIFIED KUBERNETES CLUSTER

LAUNCH TRAINING KDAPPS

- KubeDirector clusters (kdclusters) put the ML pipeline to work
- First, launch an instance of the ML training kdapp:

```
kubectl create -f cr-cluster-training-engine.yaml
```



CNCF CERTIFIED KUBERNETES CLUSTER

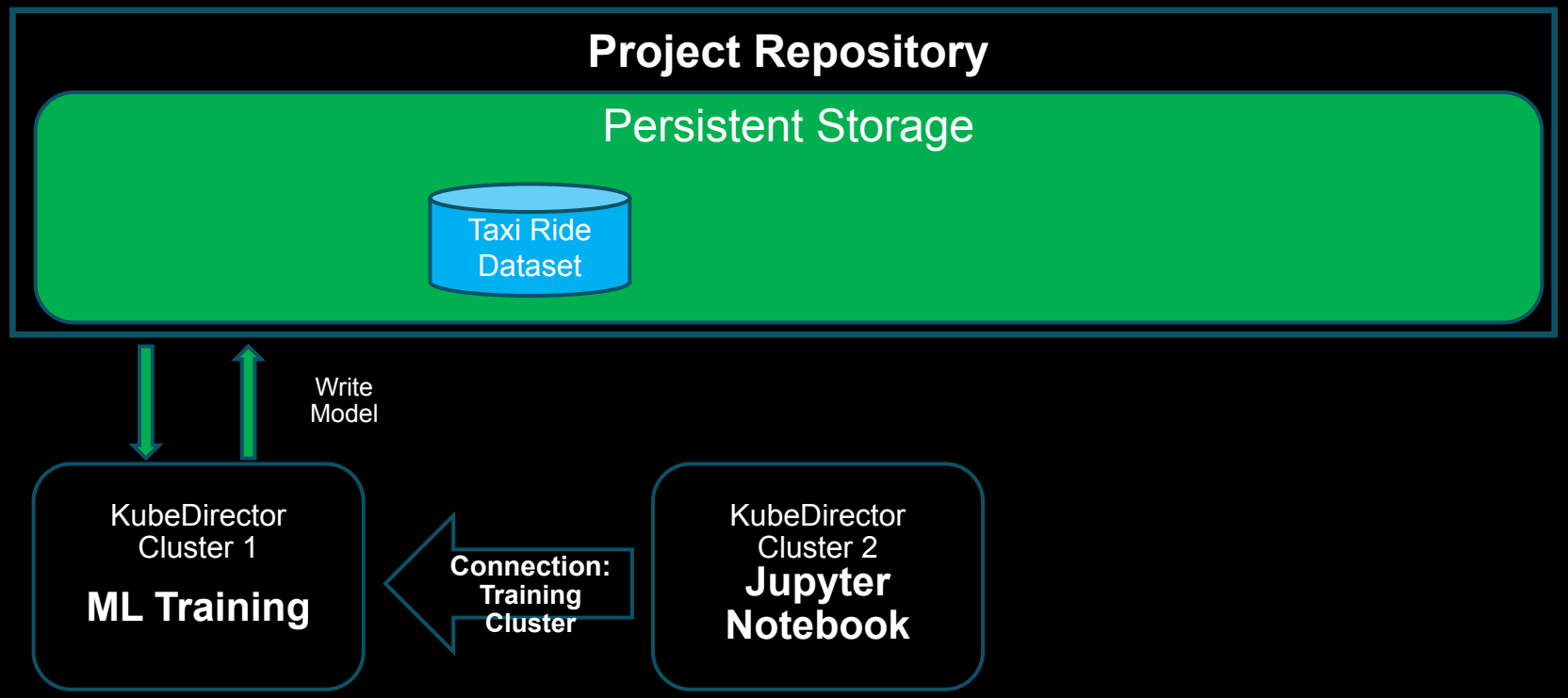
LAUNCH THE NOTEBOOK KDCLUSTER WITH A TRAINING CONNECTION

- An example Connection stanza added to our kdcluster yaml file

```
spec:
  app: “jupyter-notebook”
  appCatalog: “local”
  connections:
    clusters:
      -  “training-engine-instance”
```

- Launch an instance of a Jupyter Notebook kdcluster:

```
kubectl create -f cr-cluster-jupyter-notebook.yaml
```



CNCF CERTIFIED KUBERNETES CLUSTER

EXAMPLE JUPYTER NOTEBOOK FOR TRAINING

- %attachments magic command is used to retrieve the Training Cluster name - traningengineinstance.
- magic command - %%trainingengineinstance uses name

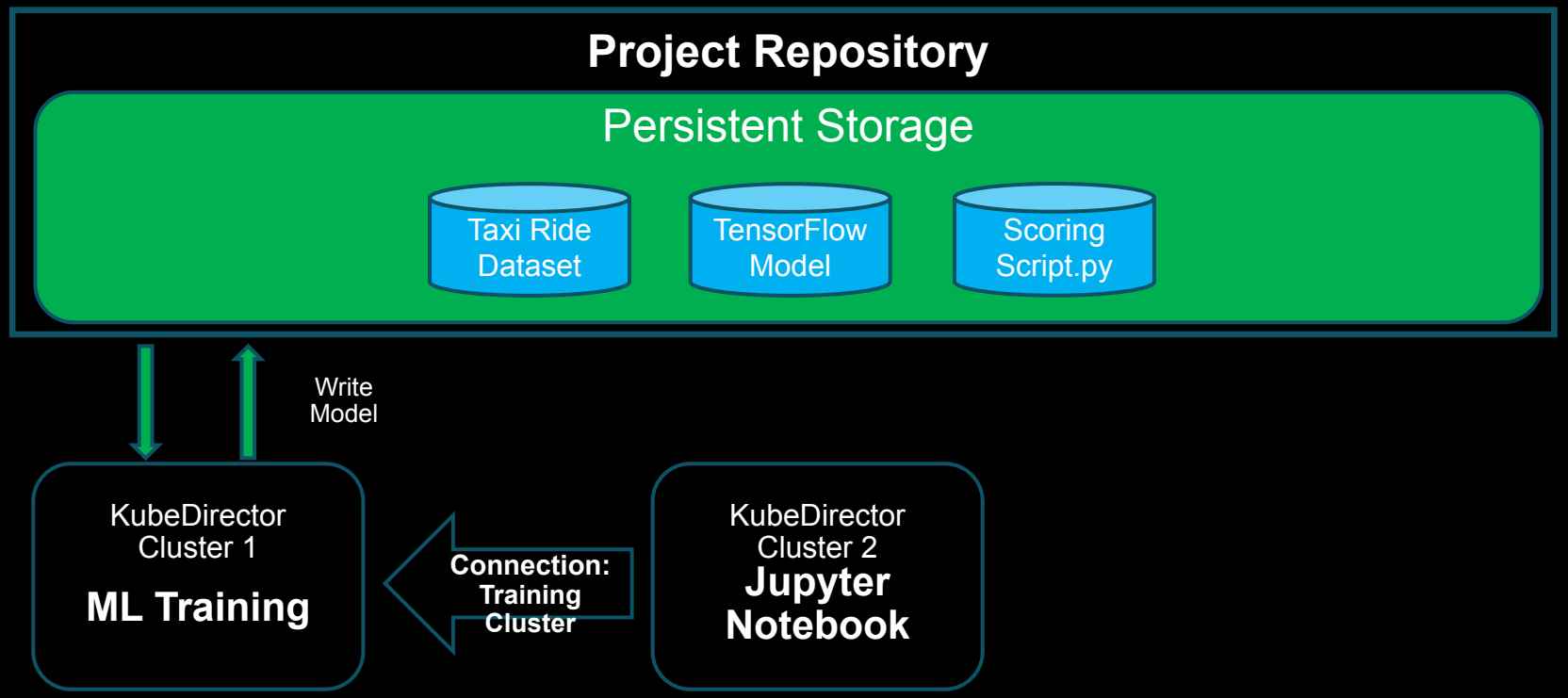
KubeDirector ML Pipeline Demo (NYC Taxi Dataset)

Prerequisite: Please ensure `lookup-ipyheader.csv` is loaded in your Project Repo under the data folder.

In [1]:
```
%attachments
```

```
Training Cluster        ML Engine
----------------------  -----------
trainingengineinstance  python
```

In [2]:
```
%%trainingengineinstance
# ----------------------------------------
# If smallDemoDataset is False, the full dataset will be downloaded from S3
# If smallDemoDataset is True, a sample dataset will be loaded from the Project Repo.
smallDemoDataset = True
# ----------------------------------------


print("Importing libraries")

import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
import tensorflow as tf
from tensorflow.keras.layers import Input, Dense, Activation,Dropout
from tensorflow.keras.models import Model
from tensorflow.keras.models import load_model
import os
import urllib
import sys


from scipy import stats
import math
import datetime


from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_squared_log_error
from math import sqrt

print("Done importing libraries")

# Specify months to train below. The available data ranges from 1/2009 to 12/2019.

# listMonthsToTrain = [[2009, 1],
#                      [2009, 2],
#                      [2009, 3],
#                      [2009, 4],
#                      [2009, 5],
#                      [2009, 6]]

listMonthsToTrain = [[x,y] for x in list(range(2009, 2020)) for y in list(range(1, 13))]


# ----------------------------------------------------------------------------------------------
baseDataUrl = "https://s3.amazonaws.com/nyc-tlc/trip+data/"

trainSetSize = 0

modelDirectory = '10yrdatasetchecknames'
```

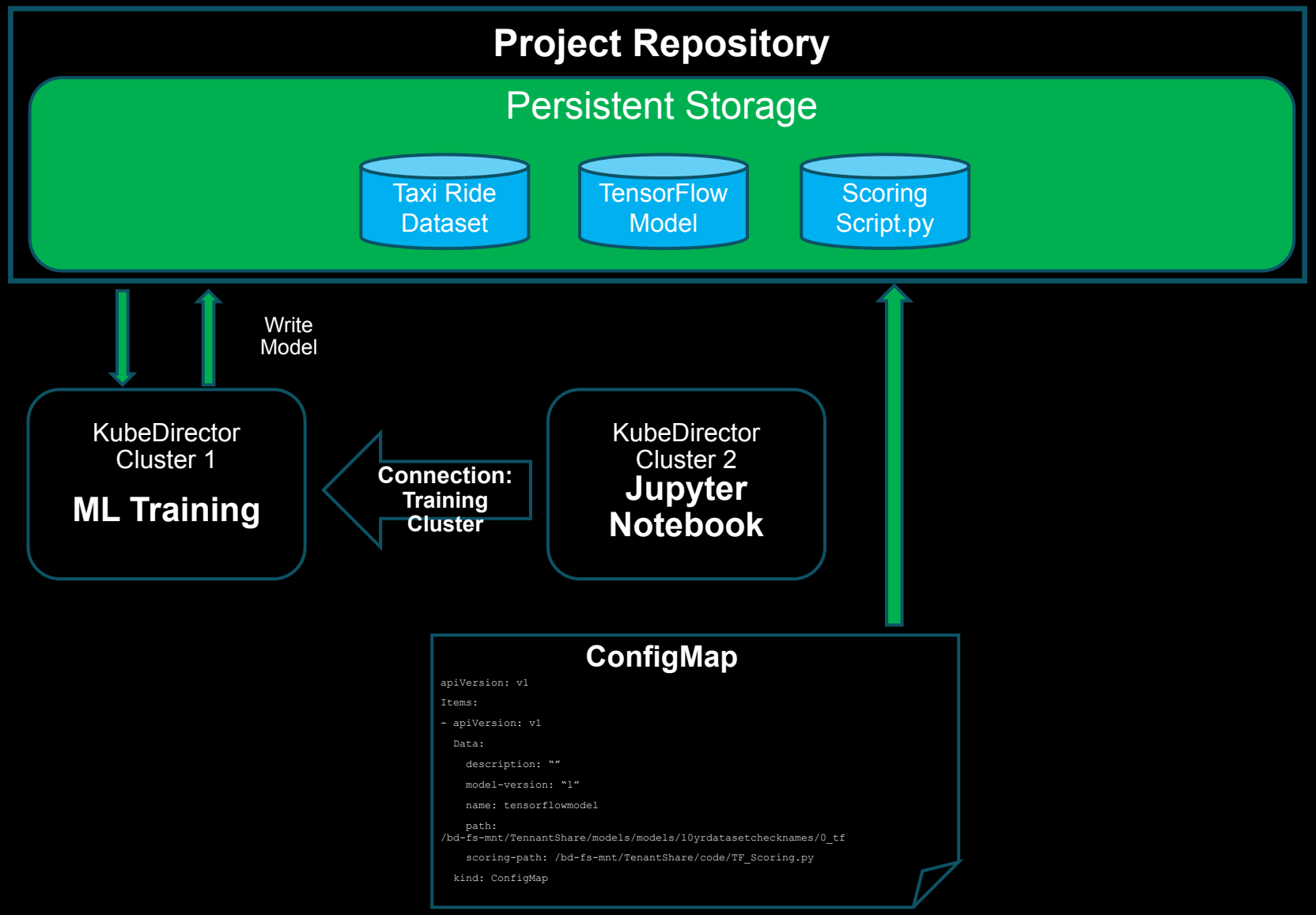


- As the training engine generates models, it will store the model data into the project repository. From the training engine's point of view, it is simply writing to a designated subdirectory of its filesystem.

THE CONFIGMAP RESOURCE

- A ConfigMap resource will store metadata about the model to be used in deployments

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: tensorflowmodel
data:
  name: tensorflowmodel
  description: "example model"
  model-version: "1"
  path: /bd-fs-mnt/TenantShare/models/10yrdatasetchecknames/0_tf
  scoring-path: /bd-fs-mnt/TenantShare/code/TF_Scoring.py
```

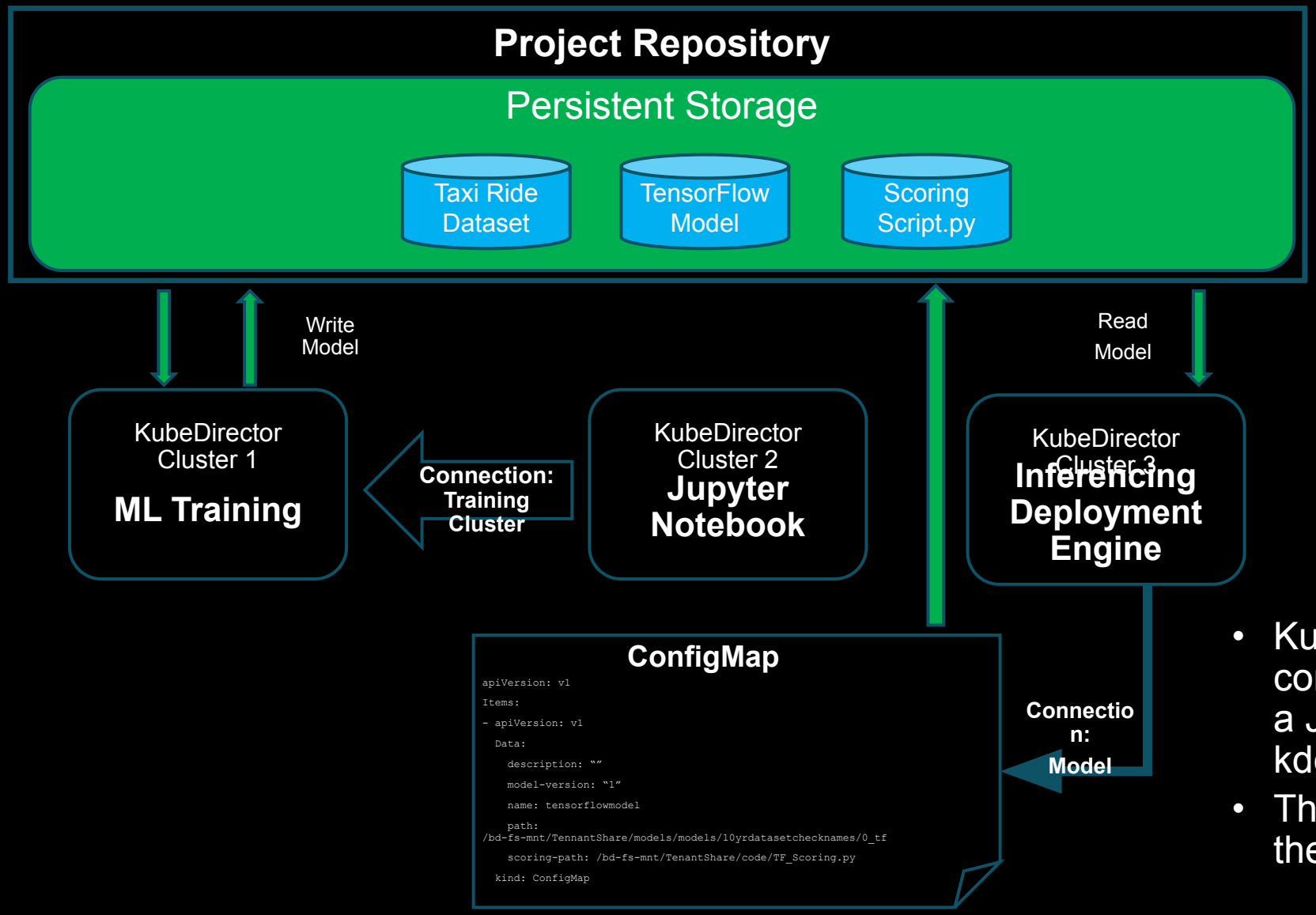


CNCF CERTIFIED KUBERNETES CLUSTER

INFERENCE SERVER KDCLUSTER -> MODEL CONFIGMAP

- For this deployment, the example cr-cluster-endpoint-wrapper.yaml file can be used
- Similar to how the Jupyter Notebook kdcluster yaml file was modified, this kdcluster yaml file will be edited to include the Connection stanza
- A new property in the top-level spec section is added

```
spec:
 app: deployment-engine
 appCatalog: “local”
 connections:
   configmaps:
   -  “tensorflowmodel”
```



- KubeDirector can inject the contents of ConfigMap into a JSON file within the kdcluster's app containers
- They can then be used by the deployment app

CNCF CERTIFIED KUBERNETES CLUSTER

ASKED AND ANSWERED

- The “haproxy” service port on the inference deployment can now be used to service REST API queries

```
Kartiks-MBP:TensorFlow kartik$ vi query_api_script_tf.py
Kartiks-MBP:TensorFlow kartik$ python3 query_api_script_tf.py
Please enter the gunicorn access point URL of your deployment cluster's RESTServer: mip-bd-vm140.mip.storage.hpecorp.net:10017
Please enter the Auth Token of your RESTServer: dcff8d05e3af7e8e428faf4c3a9a05d4
Please enter the model name as shown under the Model Registry: tensorflowmodel
Please enter the model version number: 1


Please enter 1 if the ride is during work hours (Mon-Fri, 8am to 5pm), enter 0 otherwise: 1
Please indicate the latitude of the pickup point (between 40.550 and 40.925): 40.675
Please indicate the longitude of the pickup point (between -73.750 and -75.250): -73.89
Please indicate the latitude of the dropoff point (between 40.550 and 40.925): 40.98
Please indicate the longitude of the dropoff point (between -73.750 and -75.250): -74.98
Please indicate the approx trip distance in miles: 15
Please enter 1 if the trip will occur on a weekday, 0 otherwise: 1
Please indicate what hour of day the trip occurred (range: 0 to 23): 5
The ride duration prediction is 1679.6575 seconds.
Kartiks-MBP:TensorFlow kartik$ _
```

FUTURE WORK AND MORE INFORMATION

- Enhance model concept
- Distributed TF with Slurm and GPUs
- Add model registry, dataset mgmt, feature engg
- Encrypted secrets
- **Policy for role scale**
- **Placement constraints**
- **More here -** https://github.com/bluek8s/kubedirector/issues

KUBEDIRECTOR APPLICATIONS CATALOG

cassandra	**Cassandra 3.11**
CLOUDERA	Cloudera 632
CentOS	Centos 8
KD ML Ops Deployment ToolKit	Deployment Engine
elastic	ELK 771
GitLab	GitLab
jupyterhub	Jupyter Notebook
kafka	Kafka 55
Ezmeral Data Fabric (Forme MAPR)	MapR 610

APACHE Spark	**Spark 245**
TensorFlow	TensorFlow CPU
TensorFlow	TensorFlow GPU
KD ML Ops Training Toolkit	Training Engine
ubuntu	Ubuntu 18.04

https://github.com/bluek8s/kubedirector/tree/master/deploy/example_catalog/

Many popular applications ready to deploy as KDApps

THANK YOU!

Q & A

LEARN MORE:

Blog post: Building Dynamic Machine Learning Pipelines with Kubernetes

KubeCon North America KubeDirector presentation

Twitter Handles:

Kartik Mathur: @kartik_mathur01

Don Wake: @donwaketech

Tom Phelan: @tapbluedata

Slack

http://bit.ly/KubeDirectorSlack

eMail:

Kartik Mathur: kartik.mathur@hpe.com

Don Wake: donald.wake@hpe.com

Tom Phelan: thomas.phelan@hpe.com