



**Hewlett Packard
Enterprise**

Operationalization for the Machine Learning Lifecycle live demonstration

Contents

- Overview and Introduction..... 1
 - Description and Objective..... 1
 - How to schedule and access your demo 2
 - Configuration Diagram..... 4
 - Target Audience 4
 - Requirements for Presenters..... 4
 - The HPE Advantage 4
 - HPE EPA demonstration platform Compute Blocks 5
- Demo Use Cases 6
- Operationalization for the Machine Learning Lifecycle 6
 - HPE ML Ops solution Architecture 6
 - Access HPE ML Ops demonstration platform as ml ops user 7
 - Discover HPE ML Ops platform capabilities 8
 - Create your HPE ML Ops Training cluster..... 11
 - Develop/Create Jupyter Notebooks 13
- Income Prediction Demo Scenario 16
 - Load XGB_Income.ipynb notebook 17
 - Register and Deploy the Model 18
 - Run the Income prediction using Postman 22
- Fraud Detection Demo Scenario 25
 - Create your HPE ML Ops Training cluster..... 26
 - Develop/Create Jupyter Notebooks 28
 - Register and Deploy the Model 32
 - Run the Fraud Detection prediction using Postman..... 36
- Diabetes Prediction Demo Scenario..... 39
 - Create your HPE ML Ops Training cluster..... 40
 - Develop/Create Jupyter Notebooks 42
 - Register and Deploy the Model 46
 - Run the Diabetes prediction using Postman..... 49

| | |
|--|----|
| Deploy Customized Notebook Image | 51 |
| Run the Diabetes prediction using Customized Image | 53 |
| HPE Container Platform GPU Nodes Monitoring | 56 |
| Thanks! | 57 |

Overview and Introduction

Description and Objective

This demo provides an overview of HPE Machine Learning (ML) Ops solution, powered by HPE servers and HPE Container Platform. HPE ML Ops solution supports every stage of machine learning lifecycle at enterprise scale on containerized distributed clusters — data preparation, model build, model training, model deployment, collaboration, and monitoring.

The demo environment provides access to:

- HPE Container Platform (formerly BlueData EPIC v5.0)

A CONTAINER-BASED SOLUTION FOR THE ML LIFECYCLE

Standardize processes across the ML lifecycle to build, train, deploy, and monitor machine learning models.

| | | |
|---|---|---|
| Model Build Containerized sandbox environments with a variety of machine learning and deep learning applications and tools Explore | Model Training Scalable training environments with secure access to big data Explore | Model Deployment and Monitoring Flexible and scalable endpoint deployment with end-to-end monitoring Explore |
| Collaboration Enable DevOps-like processes with code, model, and project repositories Explore | Security and Control Secure multi-tenancy with integration to enterprise authentication mechanisms Explore | Hybrid Deployment Build, train, and deploy models either on premises, in the cloud, or in a hybrid model Explore |

Proceed with caution:

HPE Container Platform (EPIC) demonstration does not include any HA and/or recovery mechanisms to protect against hardware failures and non-reversible actions.




Note: Please use Operationalization for the Machine Learning Lifecycle with HPE ML Ops recorded video as backup solution.


Operationalization for the Machine Learning Lifecycle with HPE ML Ops
Recorded Demo
Introductory
04 Mins 44 Secs

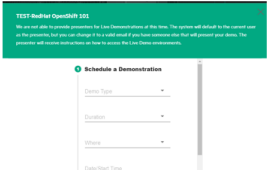
You can stop and cleanup all running cluster(s) excepted Diabetes_Prediction_Web cluster.


| | | | | | |
|--------------|-------------------------|------------------------------------|---|--|-------|
| Deployments | Diabetes_Prediction_Web | TensorFlow ModelServer | ModeServer(L/Small) | Created At: Thu May 07 2020 19:08:03 | ready |
| Data Sources | | Dependent Distro: Endpoint Wrapper | RESTServer(L/Small) LoadBalance(L/Small) | Created By: ml-ops Attached Model(s): Diabetes_Prediction_1 | |

How to schedule and access your demo

1. Request your live demo via HPE Demonstration Portal
www.hpe.com/demos/hpedemos
2. Download demonstration guide 
3. **launch** it now  **or** **Schedule** your demo 






 Gray out when demo is in use.
Need to wait or schedule for later

(*) use **Schedule** option to ensure demo availability at the time of your customer event.

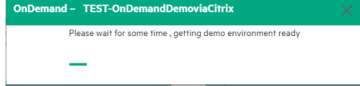

Option1: Launch demo now (not available for all demos)



Customer Event

Training / Classroom




Dry Run / Self-Paced Instruction

1. Launch your Demo after having specified demo type:
2. Demo environment is getting configured (wait a few seconds) 
3. Citrix demo session opens automatically 

* Citrix receiver plugin to be installed the first time
4. Demo is also available from HPE Demo Portal User dashboard

- Homepage
- User Dashboard
- Site Catalog
- Get Help

My Upcoming Demonstration(s)

Event ID: 13669

Event Subject: HPE ProLiant for Azure Stack Live Tour (#9009)

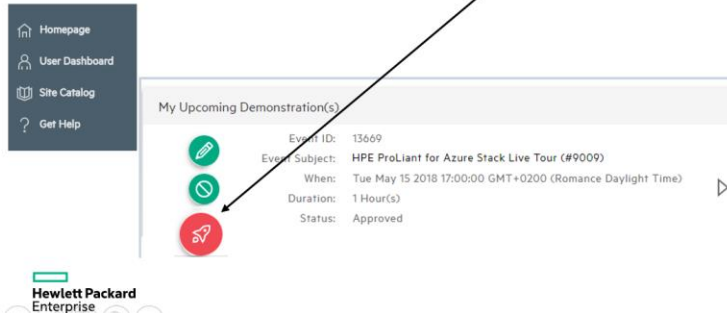
When: Tue May 15 2018 17:00:00 GMT+0200 (Romance Daylight Time)

Duration: 1 Hour(s)

Status: Approved

Option2: Scheduled a demonstration


1. Scheduled Demo will be approved and will be ready at the time you choose:
Presenter Access Instructions email is sent (15 minutes prior the demo)
with explanation how to connect.
2. Demo will directly be accessible from HPE Demo Portal User dashboard
Launch button will appear **at the time** of the demo

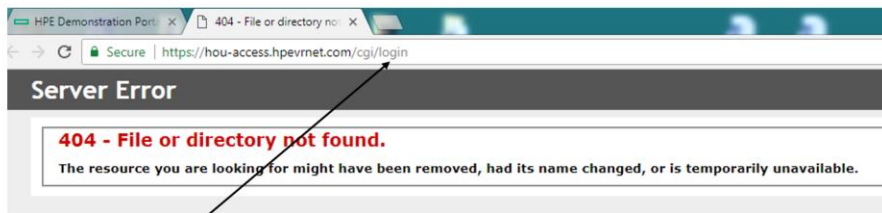


4. Run your demo (open Citrix session)
* Citrix receiver plugin to be installed the first time



In case of problem ...

When trying to launch either too early or a second time a demo via launch button  the following error may occur (**404 error** or **Invalid credentials**).



Change and run URL, replacing « login » by **logout**
<https://xxx-access.hpevrnet.com/cgi/logout>

And launch again your demo from « my upcoming demonstrations »
In HPE Demo Portal User dashboard :



4. Run your demo (open Citrix session)
* Citrix receiver plugin to be installed the first time



Optional

Configuration Diagram

Target Audience

All potential existing or new customers that are looking for solution to deploy distributed AI, machine learning (ML), and big data analytics running on Docker containers either on-premises or in the cloud..

HPE field personal and/or partners that need hands-on experience with HPE ML Ops platform will also find it valuable.

Requirements for Presenters

Presenters should have some knowledge of HPE Container Platform (EPIC) software including Elastic Platform for Big Data Analytics servers.

The HPE Advantage

HPE acquires leading provider of container-based software solution that offers an as-a-service experience for AI, machine learning, and analytics in the enterprise.. We believe this differentiated hardware EPA + software solution offer will help IT customers deploy BigData Analytics and AI in minutes to support more efficiently data science teams deriving insights from data, instead of waiting for infrastructure.

HPE EPA demonstration platform Compute Blocks

HPE Deep Learning Accelerator Block with Nvidia GPU:

HPE APOLLO 2000 GEN10 - DEEP LEARNING ACCELERATOR BLOCK

2 x HPE Apollo XL190r Gen10 - Accelerated Compute Block with GPUs



HPE Apollo 2000 Chassis



HPE ProLiant XL190r

Nvidia
Tesla V100



HPE Apollo 2000 Gen10 with 2 x XL190r

- HPE Apollo r2600 Gen10 24SFF chassis

HPE ProLiant XL190r Gen10 Server

- 2 x Intel Xeon Gold 6126 CPUs (12-core@2.1GHz)
- 2 x NVIDIA Tesla GPUs (V100)
- 384GB RAM (12x32GB)
- 2 x 960GB SSD
- 1 x 2-port 10/25GbE 640 FLR-SFP28

4 x Nvidia V100 GPU cards in a 2U frame

1
8

HPE Apollo 2000 Gen10 Standard Compute Block:

HPE APOLLO 2000 GEN10 - COMPUTE BLOCK

4 x HPE Apollo XL170r Gen10 - Standard Compute Node



HPE Apollo 2000 with 4 x XL170r

- HPE Apollo r2600 Gen10 24SFF chassis

HPE ProLiant XL170r Gen10 Server

- 2 x Intel Xeon Gold 6126 CPUs (12-core@2.1GHz)
- 384GB RAM (12x32GB)
- 2 x 480GB SSD
- 1 x 2-port 10/25GbE 640 FLR-SFP28

HPE ProLiant XL170r
384GB RAM - 2 x 480GB SSD



Demo Use Cases

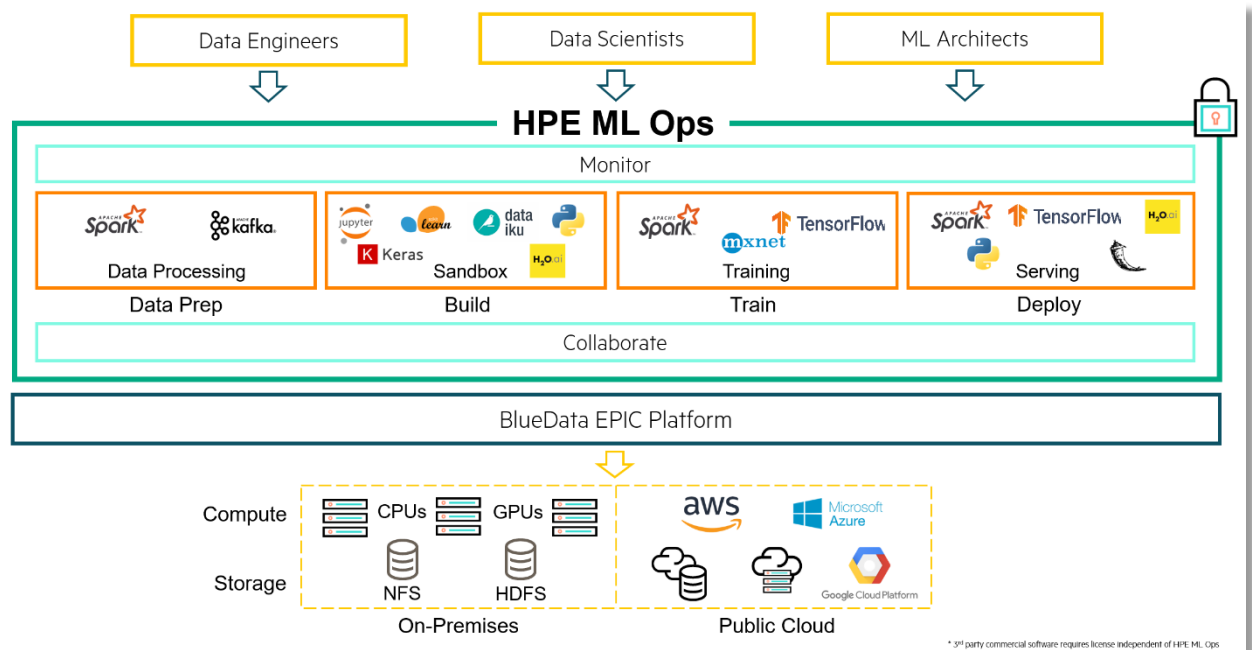
Operationalization for the Machine Learning Lifecycle

Much like pre-DevOps software development, data science organizations still spend a significant amount of time and effort when moving projects from development to production. Model version control and code sharing is manual and there is a lack of standardization on tools and frameworks, making it tedious and time-consuming to productize machine learning models.

HPE Machine Learning Ops (HPE ML Ops) extends the capabilities of the HPE Container Platform and brings DevOps-like agility to enterprise machine learning. With the HPE ML Ops solution, enterprises can implement DevOps processes to standardize their ML workflows.

HPE ML Ops provides data science teams with a platform for their end-to-end data science needs with the flexibility to run their machine learning or deep learning (DL) workloads on-premises, in multiple public clouds, or a hybrid model and respond to dynamic business requirements in a variety of use cases.

HPE ML Ops solution Architecture



Access HPE ML Ops demonstration platform as mlops user

| Environment | Web UI | Username |
|-------------|---|----------|
| HPE ML Ops | https://hpe-mlops.hpintelco.org | mlops |

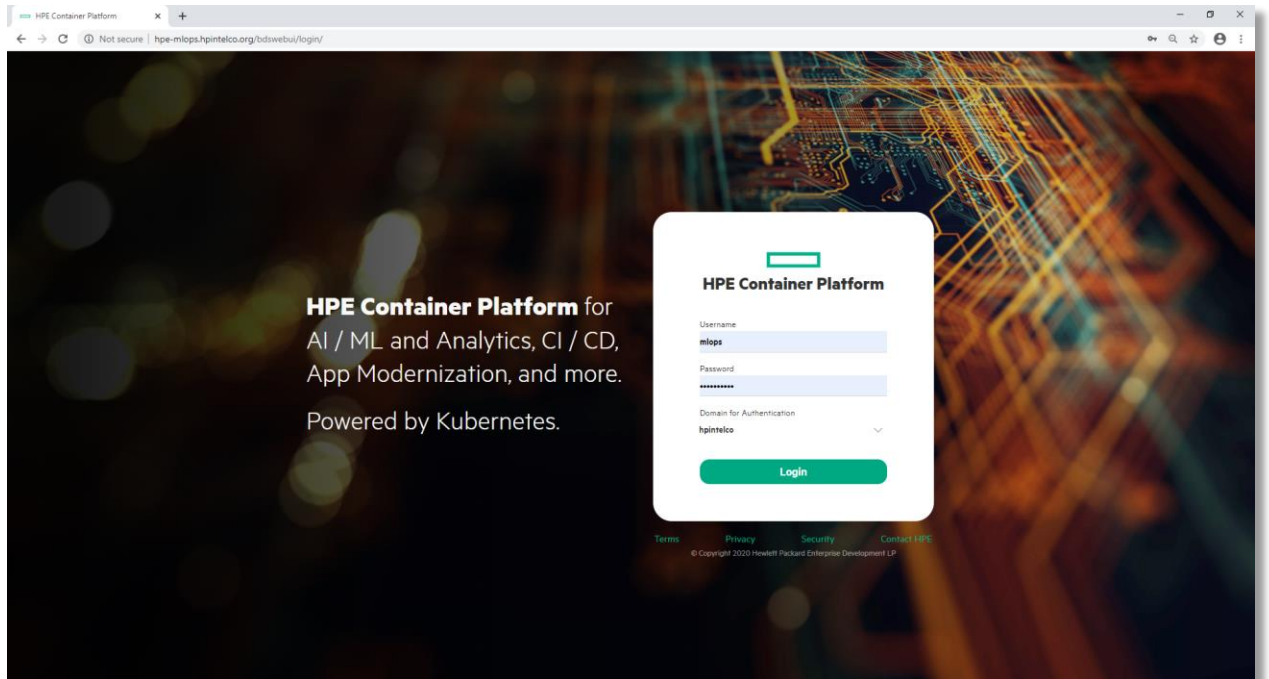
Once connected to the Jumphost station, open in your browser (chrome) the web ui login console, fill the logon credentials provided on the BlueData_EPIC html page and click "Login".



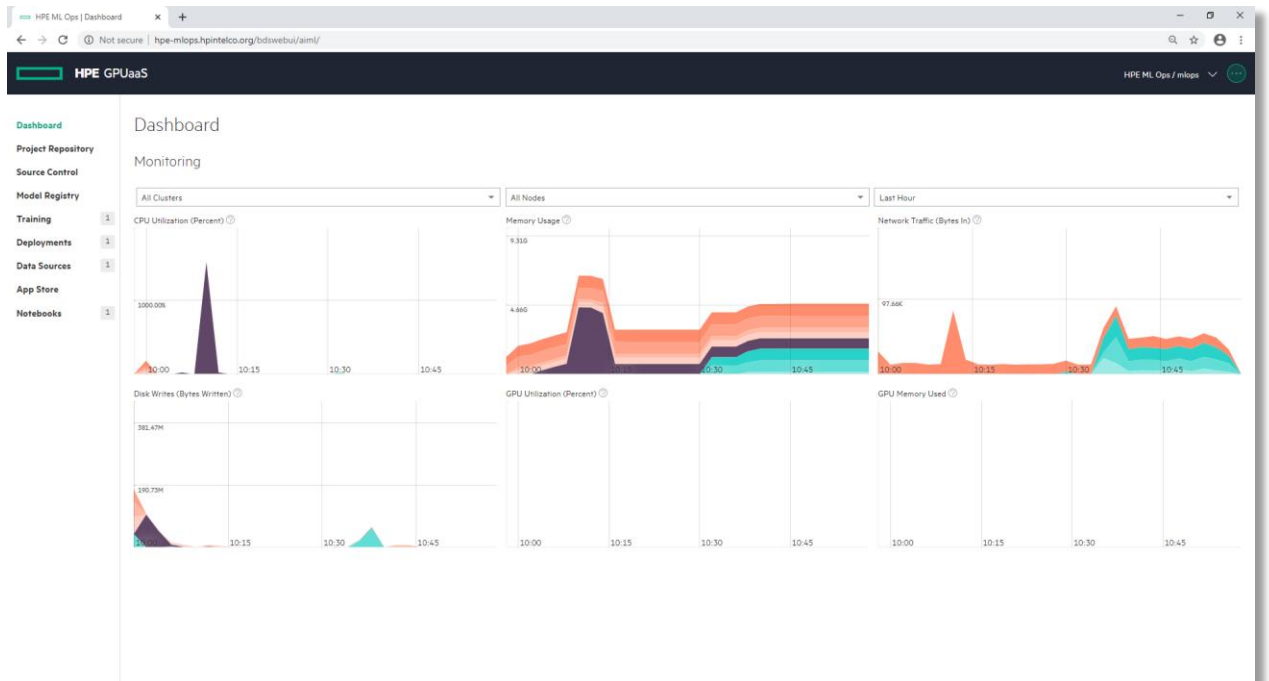
This link will start Chrome and open an html document with links and credentials!

Discover HPE ML Ops platform capabilities

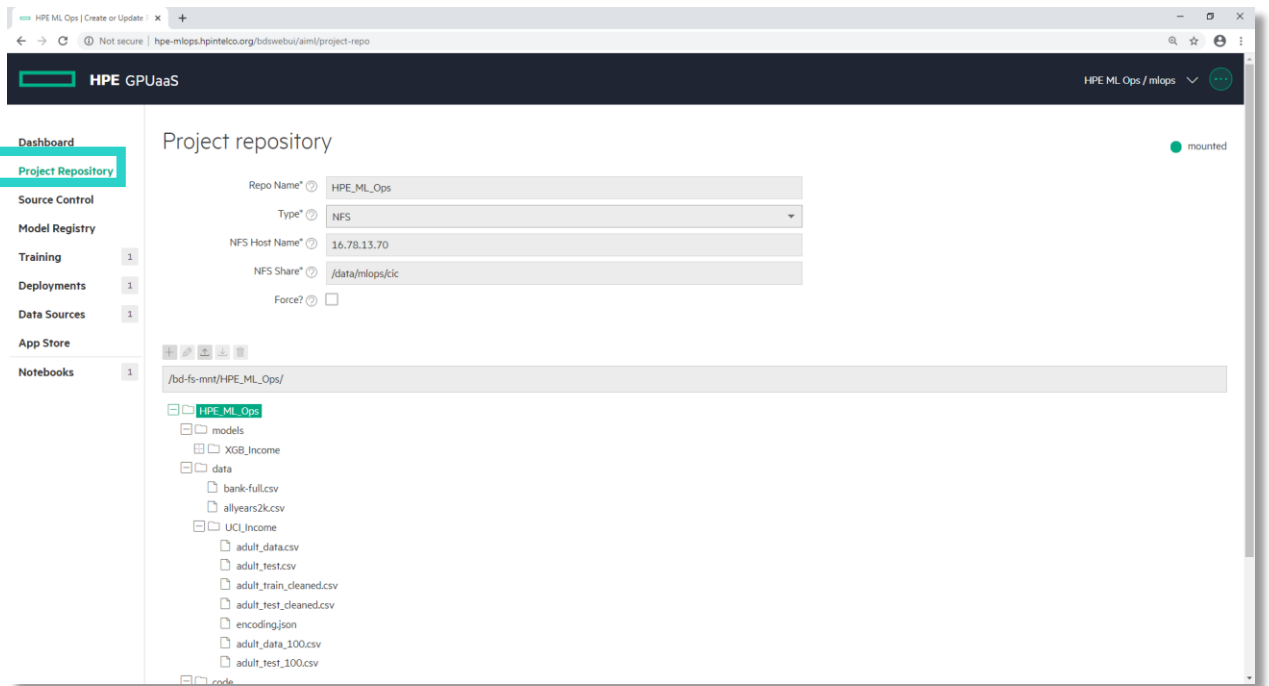
1. HPE Container Platform Login Page: Login in to the WebUI as **mlops** user: <http://hpe-mlops.hpintelco.org>



2. HPE Container Platform User Dashboard: See the monitoring

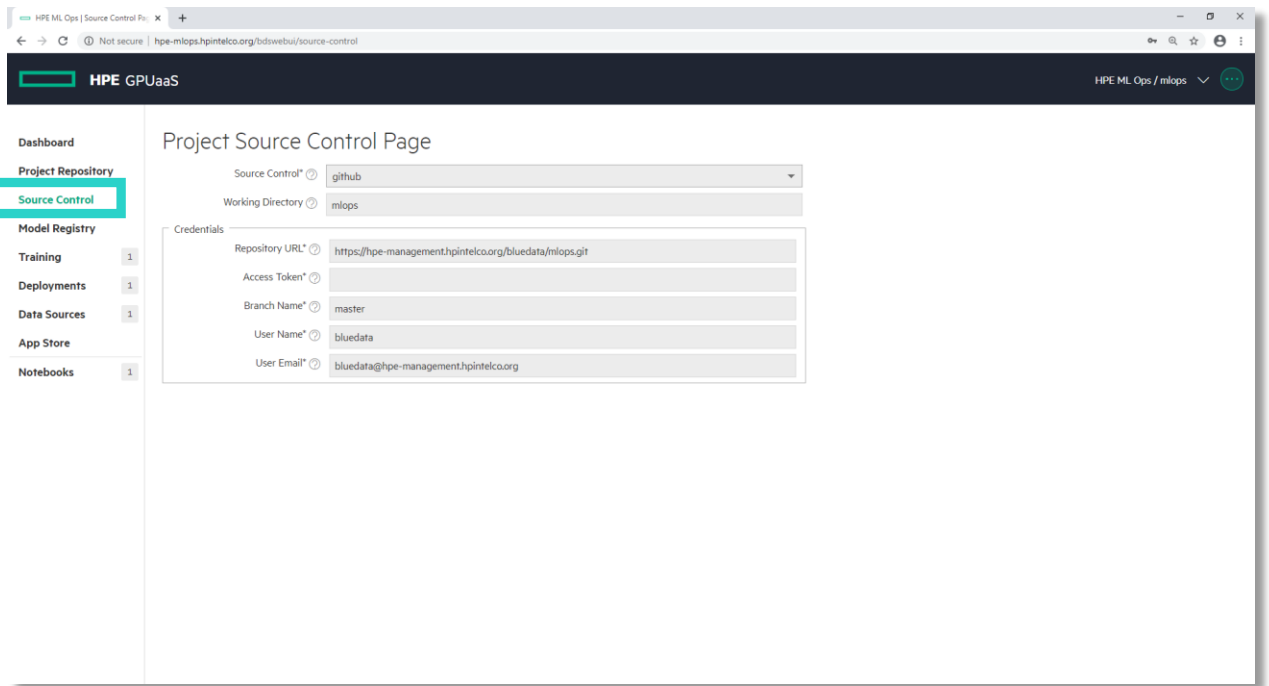


3. Select Project Repository from the left menu



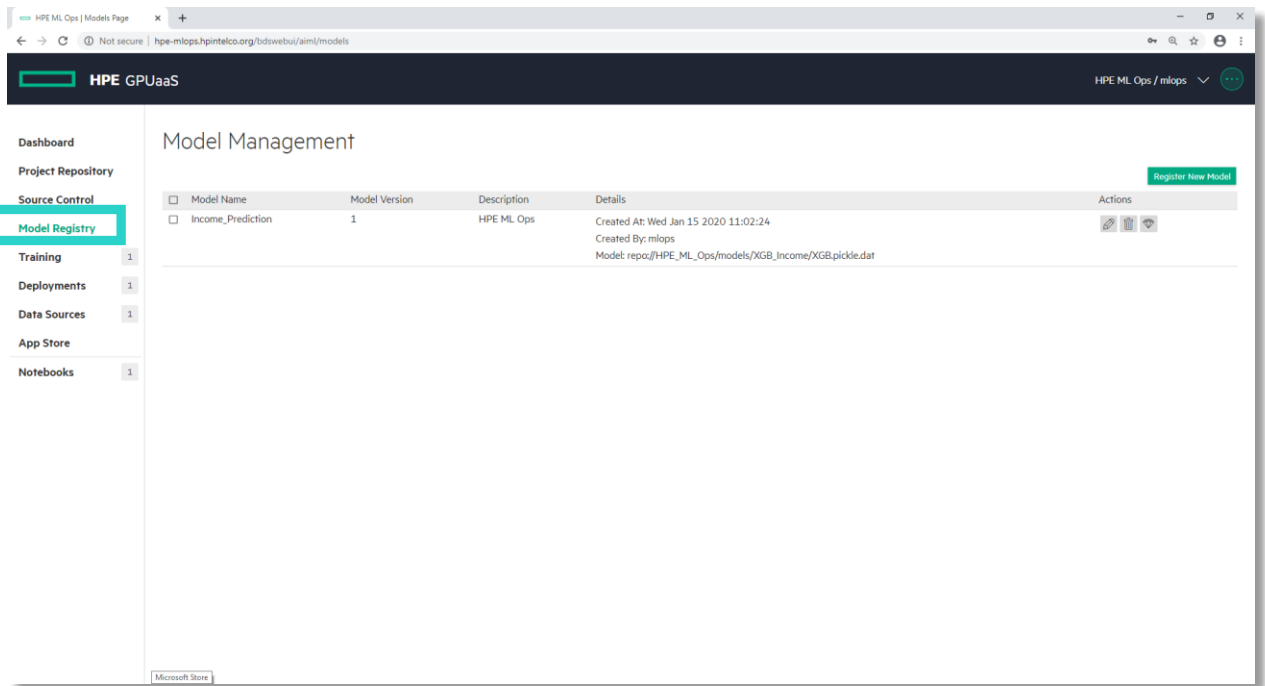
The **Project Repository** screen displays information about the project repository that was created by the Project Administrator for the current AI/ML project.

4. Select Source Control from the left menu



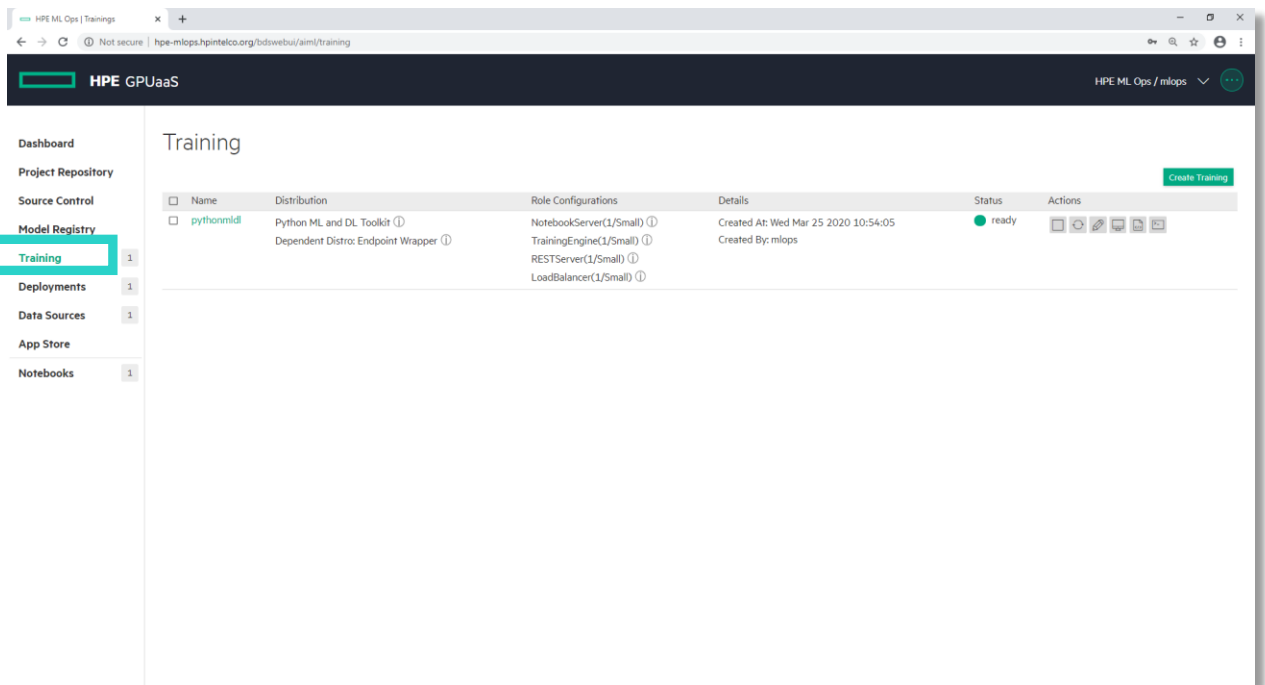
The **Project Source Control Page** screen allows you to specify a cloud-based source control service to hold notebooks for this project. Notebook clusters in this AI/ML project will retrieve notebooks from the source control location specified on this page.

5. Select Model Registry from the left menu



The **Model Management** screen displays all of the models that have been registered for the current project.

6. Select Training from the left menu



Create your HPE ML Ops Training cluster

1. Select Create Training

Training

Create Training

| Name | Distribution | Role Configurations | Details | Status | Actions |
|------|--------------|---------------------|---------|--------|---------|
|------|--------------|---------------------|---------|--------|---------|

HPE ML Ops | Create or Update x +

hpe-mlops.hpintelco.org/bdswebui/aim/create-edit-training

HPE GPUaaS HPE ML Ops / mlops

create training cluster

Cluster Detail

Name* mldemo

Description HPE ML Ops Demonstration

RunTime Image* Python ML and DL Toolkit

Node Roles

| Role | Instance Type | Count |
|----------------|-----------------------------|-------|
| NotebookServer | Small - 4 VCPU, 8.00 GB RAM | 1 |
| TrainingEngine | Small - 4 VCPU, 8.00 GB RAM | 1 |

Add-Ons

Endpoint Wrapper

| Role | Instance Type | Count |
|--------------|-----------------------------|-------|
| RETServer | Small - 4 VCPU, 8.00 GB RAM | 1 |
| LoadBalancer | Small - 4 VCPU, 8.00 GB RAM | 1 |

Upload Scaling Policy Select a valid JSON file

Advanced Settings

Don't forget to add a Training Engine and update the RunTime Image and Click on Submit

HPE ML Ops | Create or Update x +

hpe-mlops.hpintelco.org/bdswebui/aim/create-edit-training

HPE GPUaaS HPE ML Ops / mlops

create training cluster

Cluster Detail

Name* mldemo

Description HPE ML Ops Demonstration

RunTime Image* Python ML and DL Toolkit

Node Roles

| Role | Instance Type | Count |
|----------------|-----------------------------|-------|
| NotebookServer | Small - 4 VCPU, 8.00 GB RAM | 1 |
| TrainingEngine | Small - 4 VCPU, 8.00 GB RAM | 1 |

Add-Ons

Endpoint Wrapper

| Role | Instance Type | Count |
|--------------|-----------------------------|-------|
| RETServer | Small - 4 VCPU, 8.00 GB RAM | 1 |
| LoadBalancer | Small - 4 VCPU, 8.00 GB RAM | 1 |

Upload Scaling Policy Select a valid JSON file

Advanced Settings

2. Check that the cluster is ready

Cluster Overview for 'mldemo':

- Name: mldemo
- Distribution: Python ML and DL Toolkit
- Dependent Distro: Endpoint Wrapper
- Roles: NotebookServer(1/Small), TrainingEngine(1/Small), RESTServer(1/Small), LoadBalancer(1/Small)
- Created At: Wed Mar 25 2020 12:05:45
- Created By: mlops
- Status: ready

3. Open Training Cluster details (Click on mldemo)

Cluster Details for 'mldemo':

- Name: mldemo
- Distribution: Python ML and DL Toolkit
- Dependent Distro: Endpoint Wrapper
- Roles: NotebookServer(1/Small), TrainingEngine(1/Small), RESTServer(1/Small), LoadBalancer(1/Small)
- Created At: Wed Mar 25 2020 12:05:45
- Created By: mlops
- Status: ready

4. Show the cluster nodes and services. (NotebookServer / TrainingEngine / ...)

Node(s) Info Table:

| Name | Distribution | Role | Instance IP |
|----------------------|--------------------------|----------------|-------------|
| bluedata-424.bdlocal | Python ML and DL Toolkit | TrainingEngine | 172.18.0.27 |
| bluedata-423.bdlocal | Python ML and DL Toolkit | NotebookServer | 172.18.0.28 |
| bluedata-426.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.26 |
| bluedata-425.bdlocal | Endpoint Wrapper | RESTServer | 172.18.0.25 |

Services Pop-up:

- ssh: g3str3tc9-c.hpintelco.org -p 10025
- Standalone Jupyterhub: ssh: g3str3tc9-c.hpintelco.org -p 10024
- Model serving request balancer stats: http://g3str3tc9-c.hpintelco.org:10029 [Auth Token]
- Training API Server: http://g3str3tc9-c.hpintelco.org:10030/train [Auth Token]
- API Server: http://g3str3tc9-c.hpintelco.org:10026 [Auth Token]
- SSH: g3str3tc9-c.hpintelco.org -p 10027

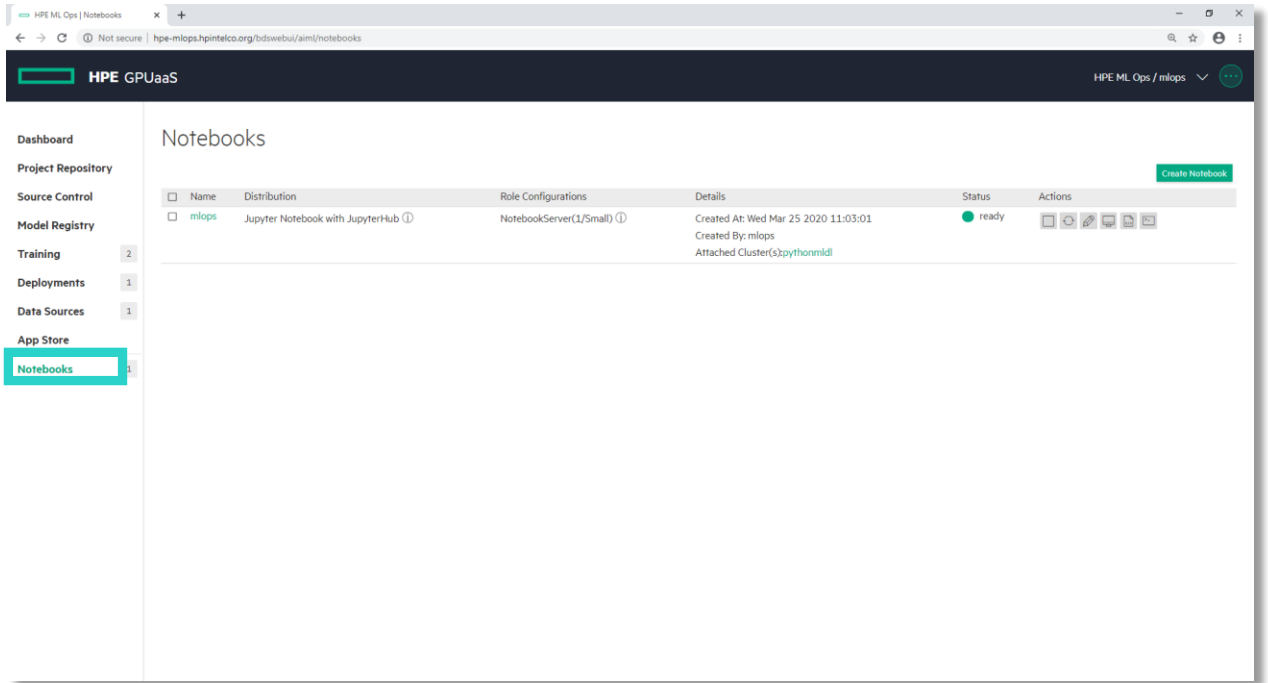
5. Check the Services Status of your cluster (Application / System services)

ServiceStatus Table:

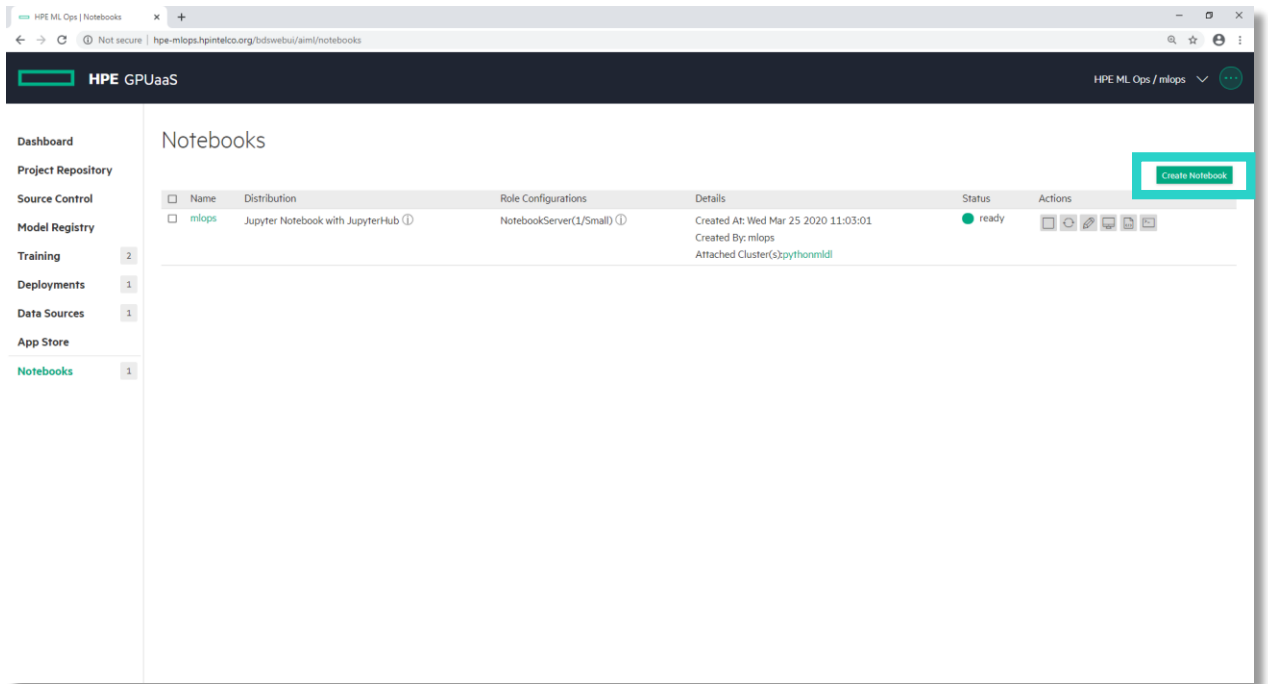
| Name | API Server | Model Serving | LoadBalancer | Standalone Jupyterhub | BlueData Agent | User Auth |
|----------------------|------------|---------------|--------------|-----------------------|----------------|-----------|
| bluedata-426.bdlocal | ● | ● | | ● | ● | ● |
| bluedata-425.bdlocal | ● | ● | | ● | ● | ● |
| bluedata-423.bdlocal | ● | ● | | ● | ● | ● |
| bluedata-424.bdlocal | ● | ● | | ● | ● | ● |

Develop/Create Jupyter Notebooks

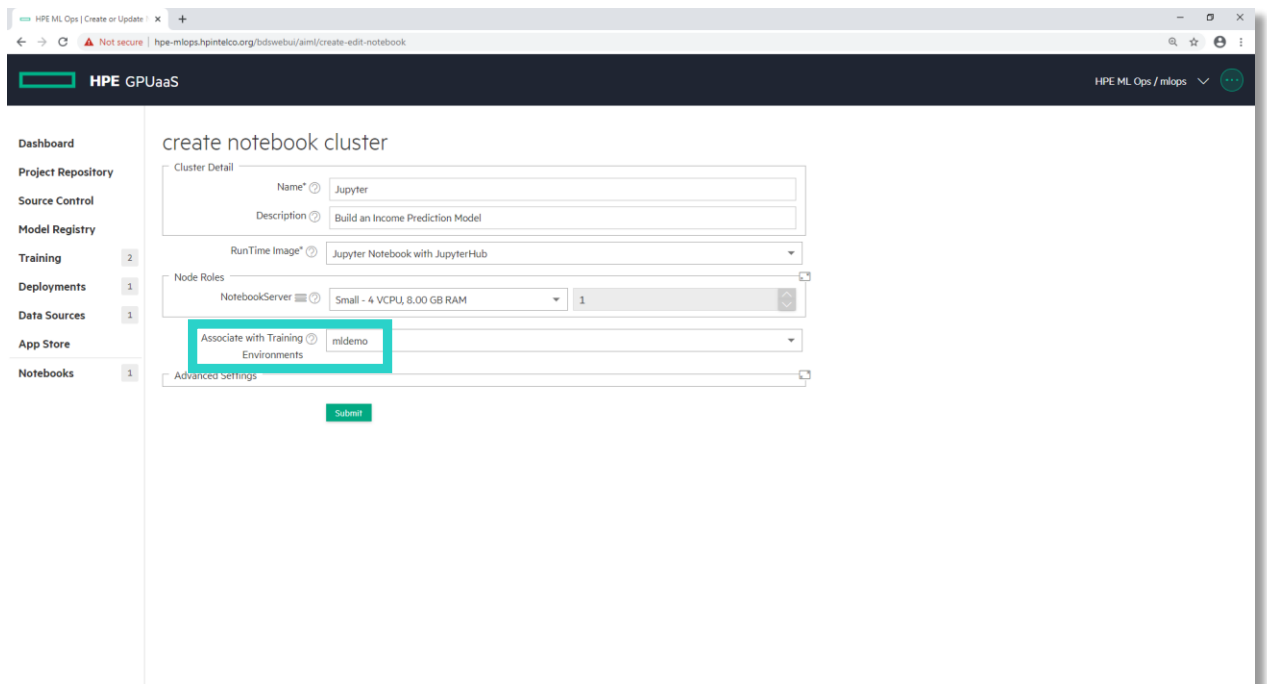
1. Select Notebooks from the left menu



2. Create a Jupyter Notebook (Click on Create Notebook)

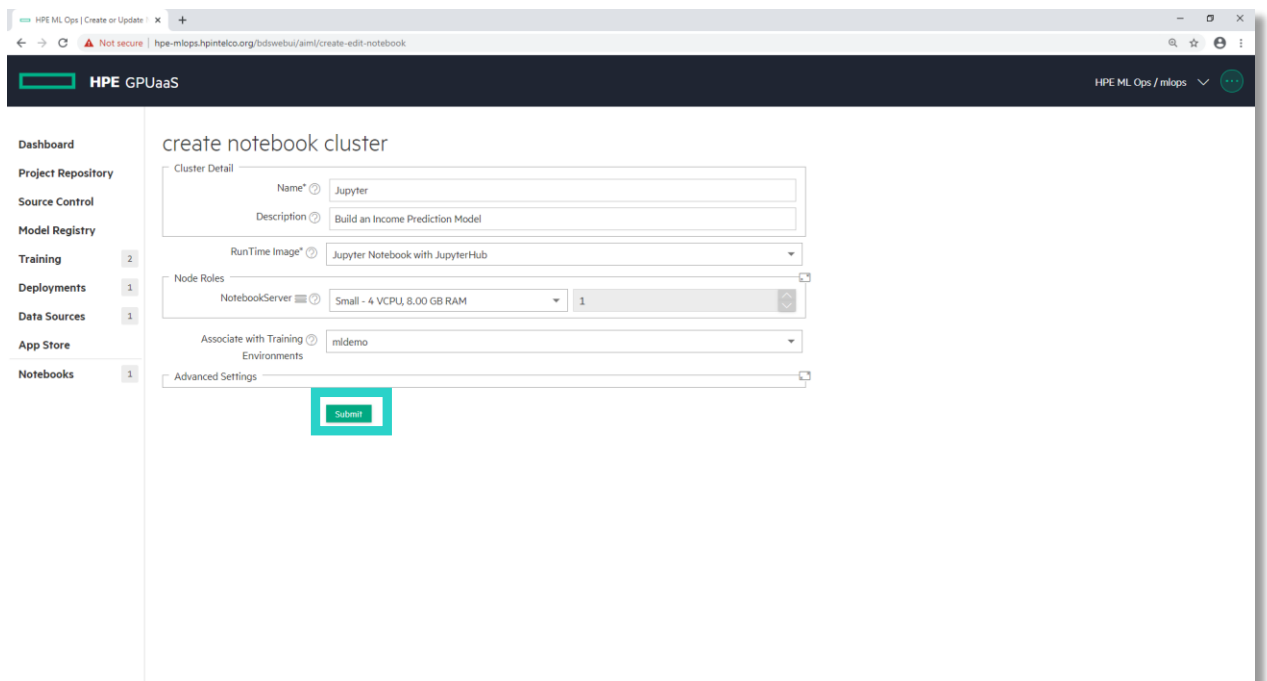


3. Associate the Notebook with your Training cluster

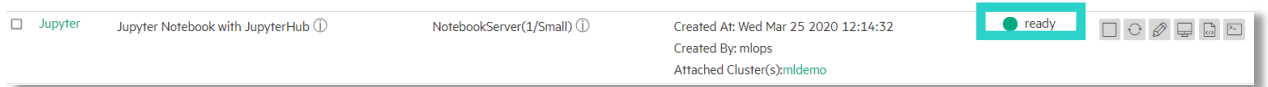


Don't forget to select Training Engine, setup flavor and update the Jupyter RunTime Image

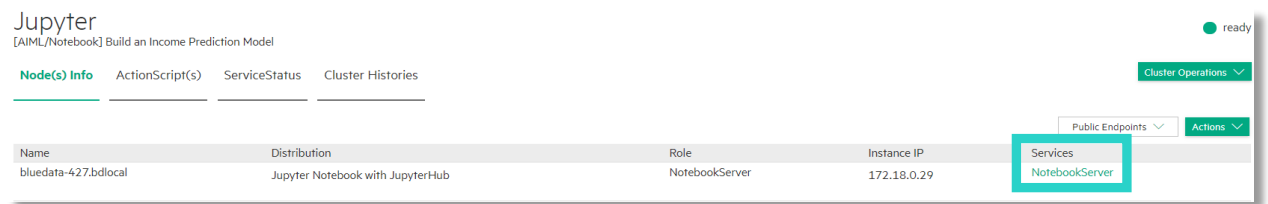
4. Create Notebook Cluster (Click on Submit)



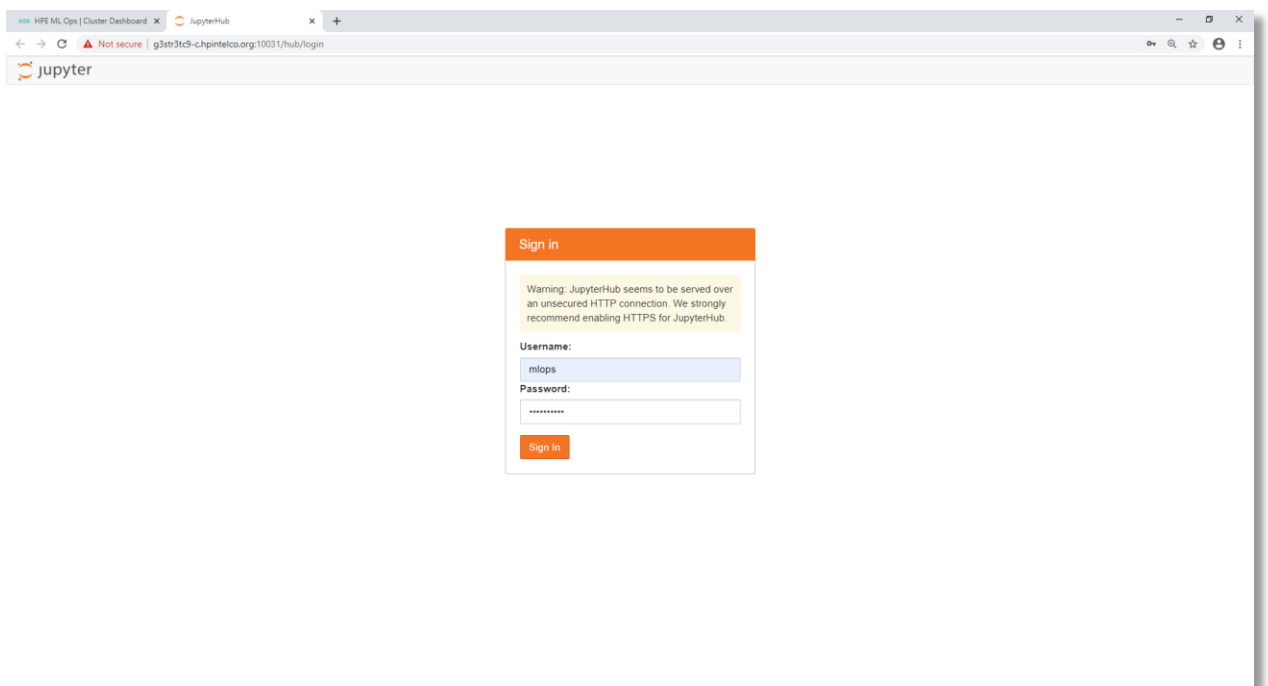
5. Make sure that the status of your notebook is ready



6. Open JupyterHub web ui (Click on NotebookServer)



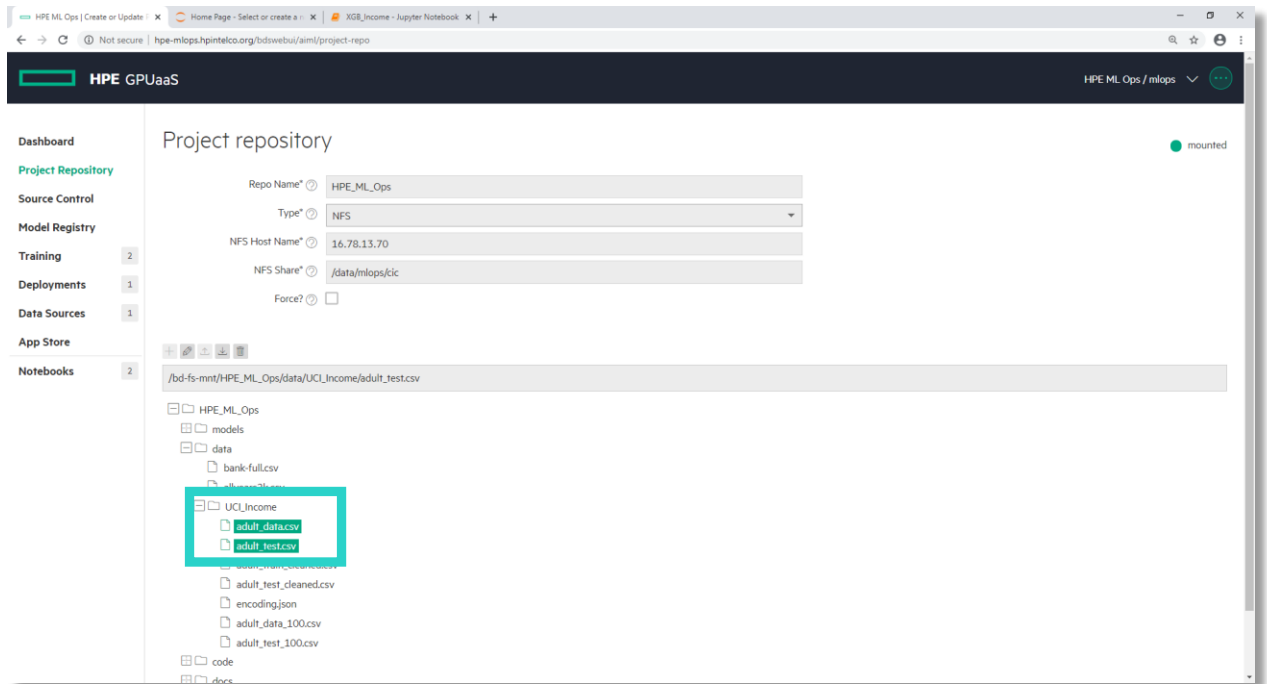
7. Login in to the Jupyter WebUI as mllops user



Income Prediction Demo Scenario

This tutorial uses a model to classify a person's income as being either less than or equal to \$50,000 or more than \$50,000.

Data set is available from <https://archive.ics.uci.edu/ml/datasets/Adult>. This data set is a spreadsheet with approximately 32,000 rows of training data that was acquired from the 1994 Census database.



Data set is in the Project Repository

The features (columns) in this spreadsheet that are used to train the model are:

- age
- workclass
- fnlwgt (the number that the census believes represents the population)
- education
- education_num (number representation of education)
- marital_status
- occupation
- relationship
- race
- sex
- capital_gain
- capital_loss
- hours_per_week
- native_country

The last column indicates the income classification of that individual.

Load XGB_Income.ipynb notebook

1. Load XGB_Income.ipynb notebook

Open the `XGB_Income.ipynb` file, and then run each cell individually. This generates cleaned `.csv` files, an encoded file, and model files. The Notebook contains detailed comments and explanations.

```
In [1]: %Attachments
Training Cluster ML Engine
-----
!demo python

In [2]: %%!demo
print('test')

History URL: http://bluedata-425.bdlocal:10001/history/1

In [3]: !logs --url http://bluedata-425.bdlocal:10001/history/1

Job Status: Finished
test

In [4]: import numpy as np
import pandas as pd
import os
import json
import seaborn as sns
sns.set(font_scale=1.5)

%matplotlib inline

In [5]: def saveInProjectRepo(path):
ProjectRepo = os.popen('bdvc11 --get cluster.project_repo').read().rstrip()
return str(ProjectRepo + '/' + path)
```

(Caution: Update the variable according to your Training Cluster including History urls.)

Once the model is tuned to the optimal parameters, run the cell on a remote Training cluster (where there are potentially more resources to train a model on a larger dataset).

```
In [79]: %%demo
# Importing libraries
print("Importing libraries")
import numpy as np
import pandas as pd
import os
import pickle
import xgboost as xgb
import datetime
from sklearn.model_selection import GridSearchCV

# Start time
print("Start time: ", datetime.datetime.now())

# Project repo path function
def saveInProjectRepo(path):
    ProjectRepo = os.popen('bdvcli --get cluster.project_repo').read().rstrip()
    return str(ProjectRepo + '/' + path)

# Reading in data
print("Reading in data")
train = pd.read_csv(saveInProjectRepo('data/UCI_Income/adult_train_cleaned.csv'))
print("Done reading in data")

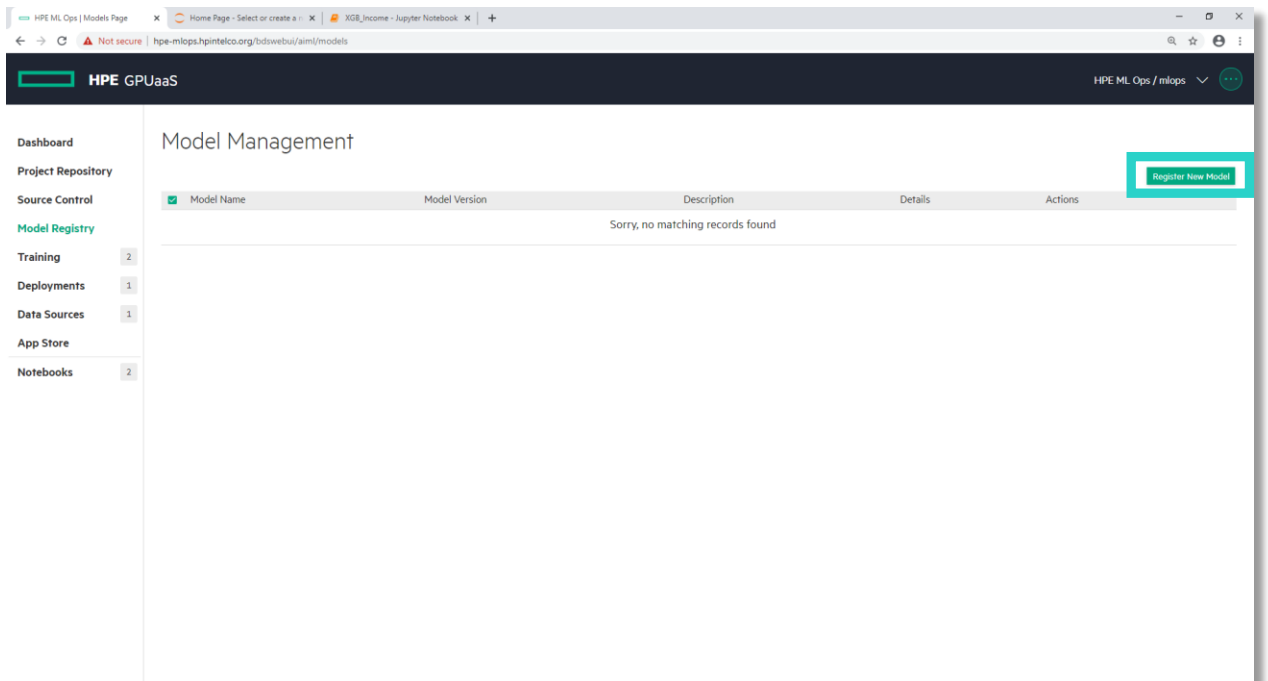
# Extracting target values
y_train = train.pop('wage_class')

# Model development / Training
print("Training...")
xgdmatrix = xgb.DMatrix(train, y_train)
our_params = {'eta': 0.1, 'seed': 0, 'subsample': 0.8, 'colsample_bytree': 0.8, 'objective': 'binary:logistic',
              'max_depth': 3, 'min_child_weight': 1}
cv_xgb = xgb.cv(params=our_params, dtrain=xgdmatrix, num_boost_round=3000, metrics=["error"],
                early_stopping_rounds=100)
optimal_rounds = len(cv_xgb)
final_gb = xgb.train(our_params, xgdmatrix, num_boost_round = optimal_rounds)

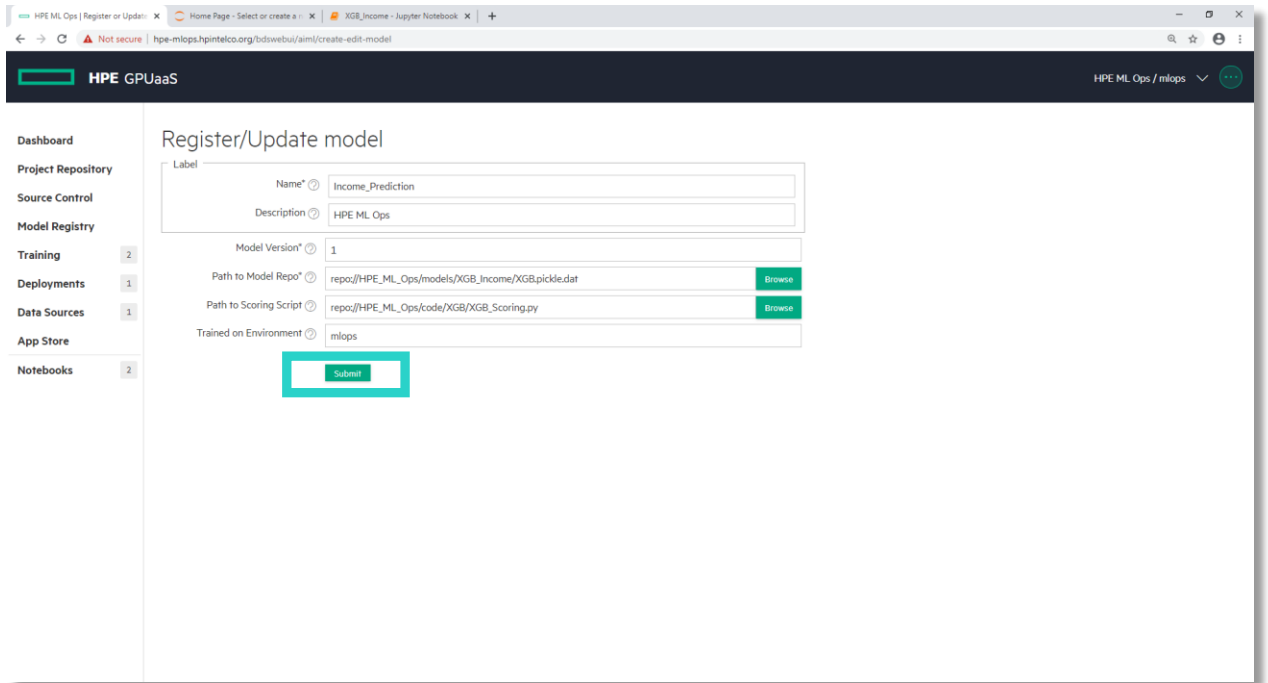
# Save model into project repo
print("Saving model")
final_gb.save_model(saveInProjectRepo('models/XGB_Income/XGB.model'))
```

Register and Deploy the Model

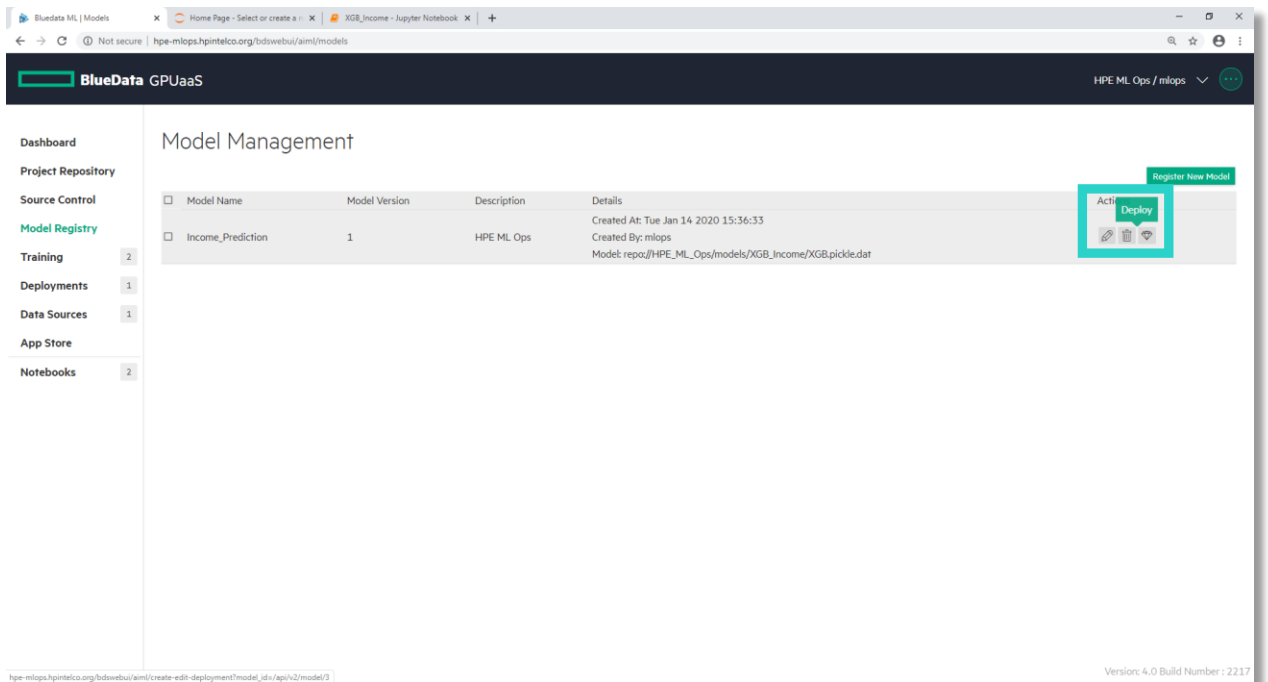
1. Select Model Registry from the left menu and click on Register New Model



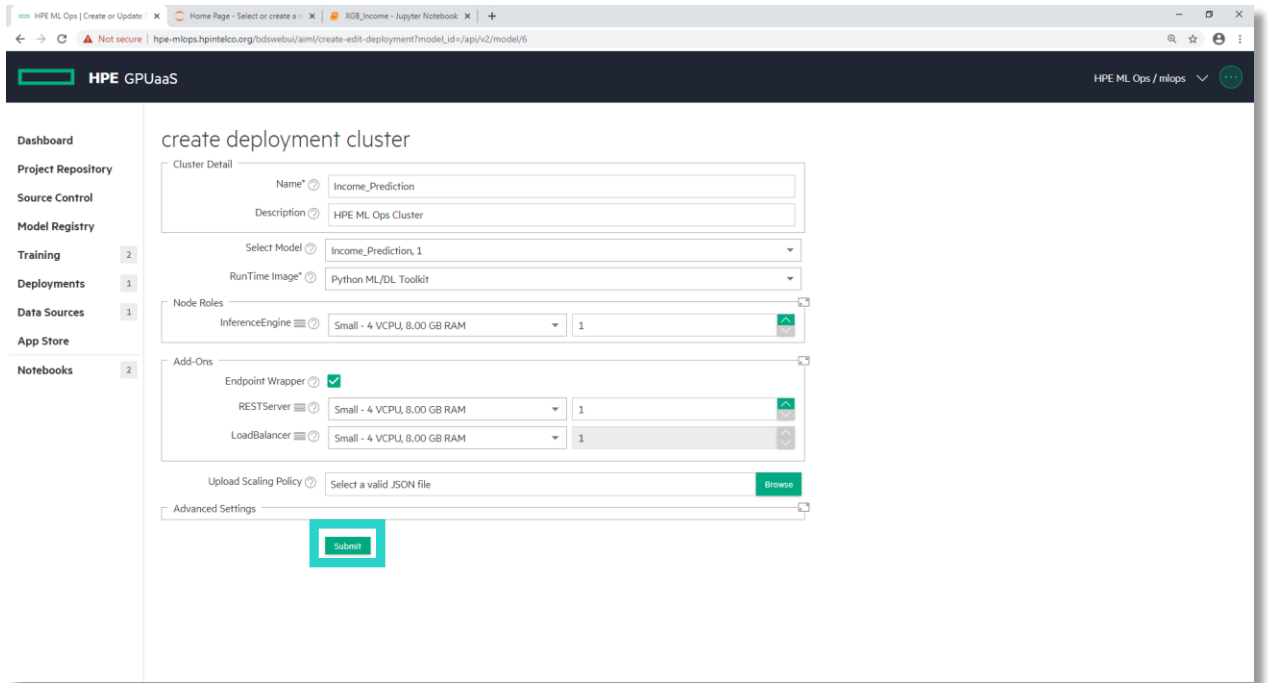
2. Register New Model using the information displayed below.



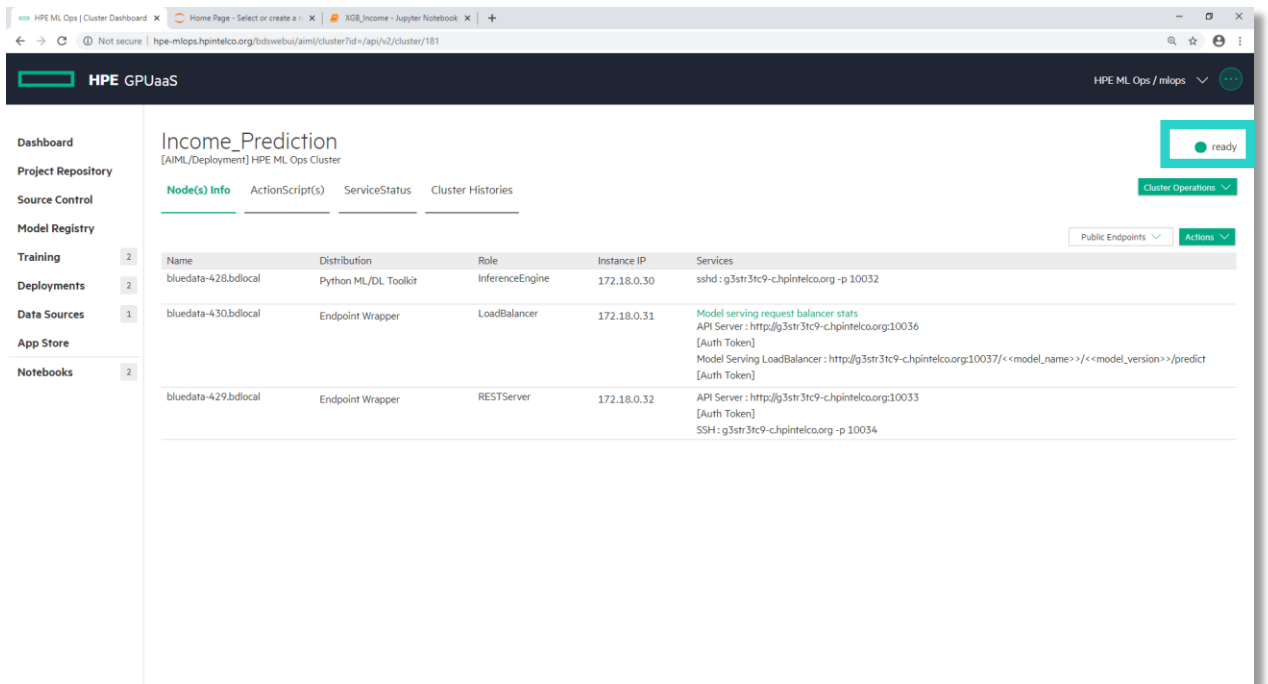
3. Deploy your registered model (Click on Deploy)



4. Create a Deployment Cluster for Income Prediction (Click on Submit)



5. Income Prediction Cluster is now Ready



6. Generate your Prediction Request

Income_Prediction
[AIML/Deployment] HPE ML Ops Cluster

ready

Cluster Operations

Node(s) Info ActionScript(s) ServiceStatus Cluster Histories

Public Endpoints Actions

| Name | Distribution | Role | Instance IP | Services |
|----------------------|----------------------|-----------------|-------------|---|
| bluedata-428.bdlocal | Python ML/DL Toolkit | InferenceEngine | 172.18.0.30 | sshd : g3str3tc9-c.hpintelco.org -p 10032 |
| bluedata-430.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.31 | Model serving request balancer stats API Server : http://g3str3tc9-c.hpintelco.org:10036 [Auth Token] LoadBalancer : http://g3str3tc9-c.hpintelco.org:10037/<<model_name>>/<<model_version>>/predict |
| bluedata-429.bdlocal | Endpoint Wrapper | RESTServer | 172.18.0.32 | API Server : http://g3str3tc9-c.hpintelco.org:10033 [Auth Token] SSH : g3str3tc9-c.hpintelco.org -p 10034 |

Copy the link and check update the port + model version

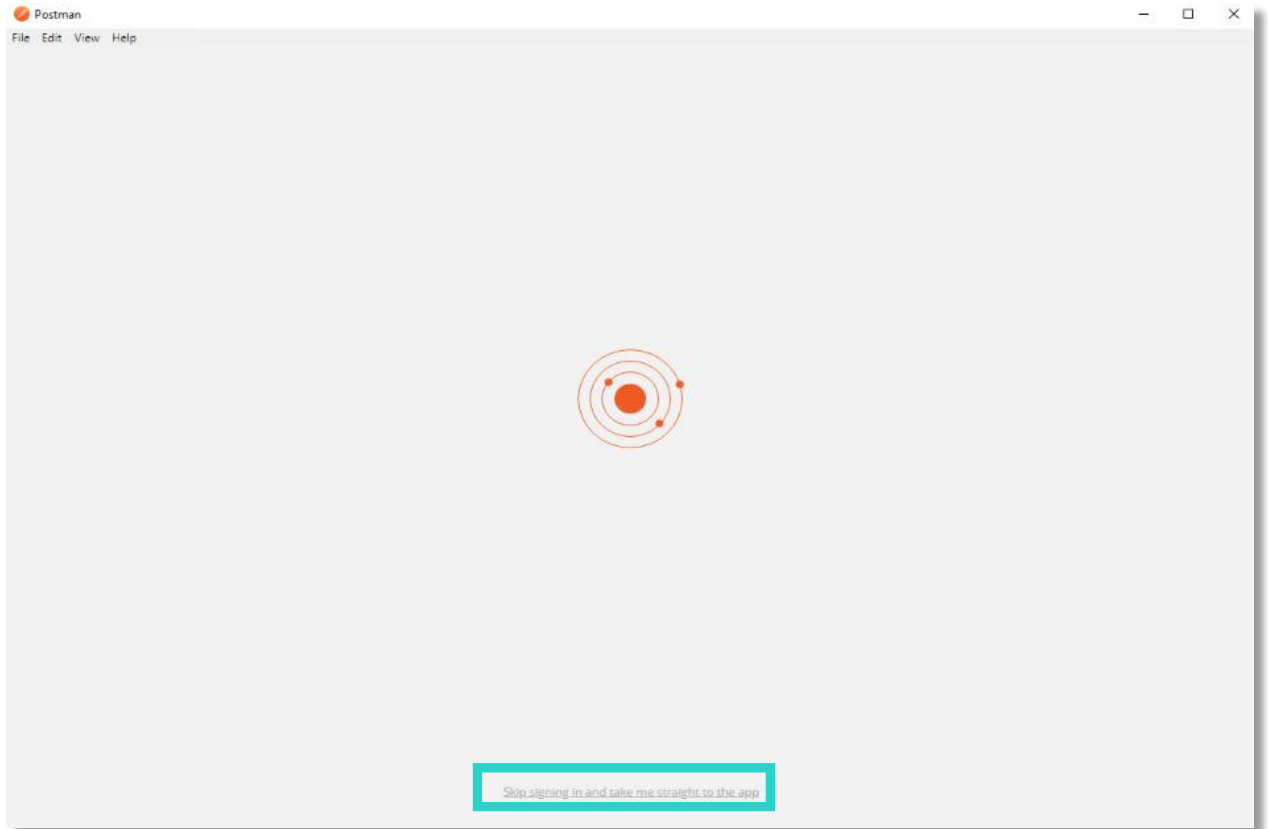
http://g3str3tc9-c.hpintelco.org:10037/Income_Prediction/1/predict
[Auth Token] Click to copy the token

Run the Income prediction using Postman

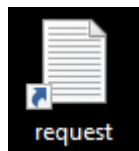


This link will start Postman!

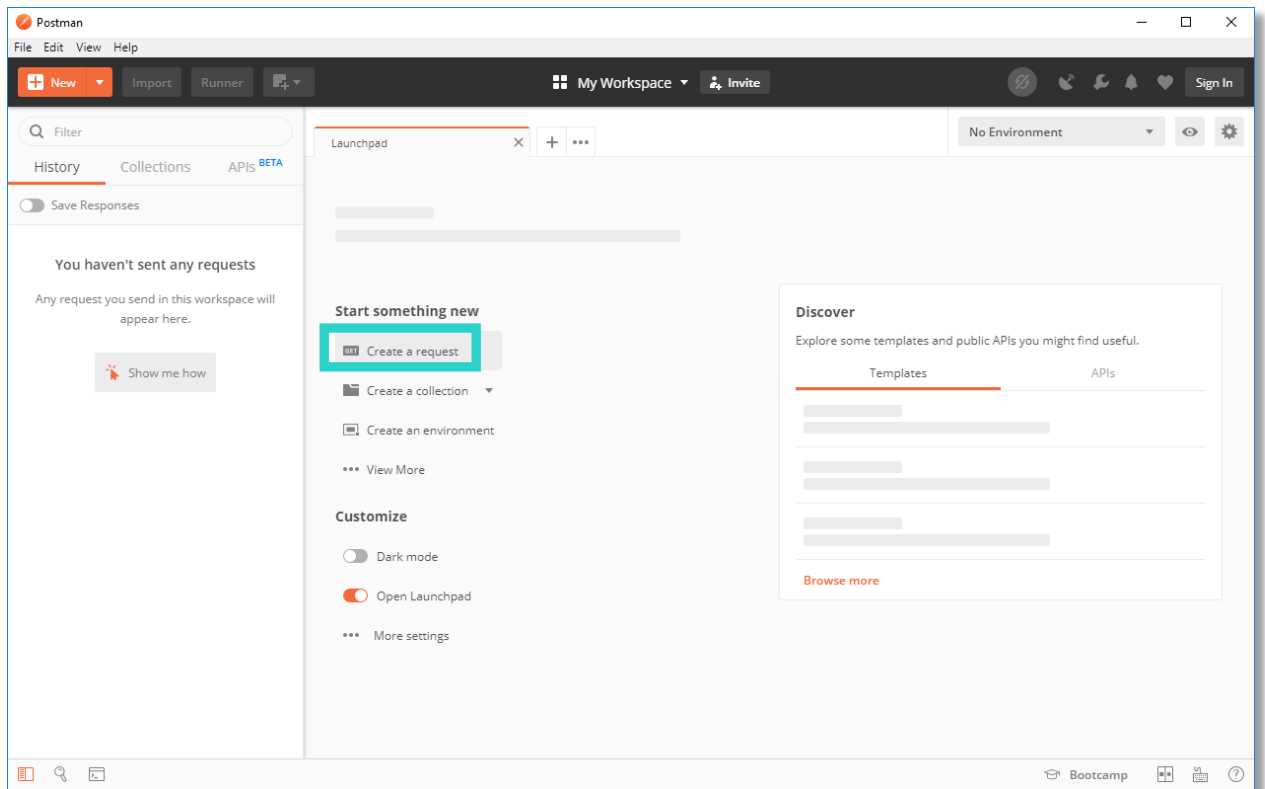
1. Select Skip signing in to go directly to the app



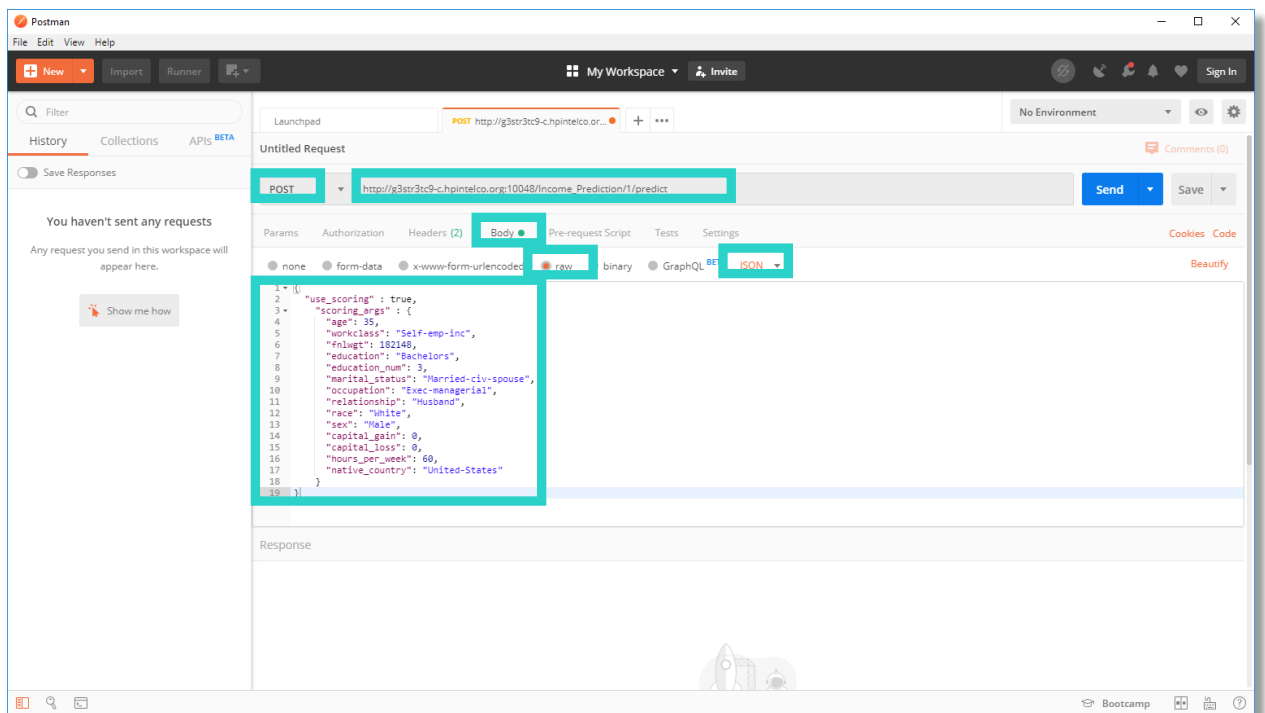
2. Copy the Income Json request in Postman body tab: Click on below icon



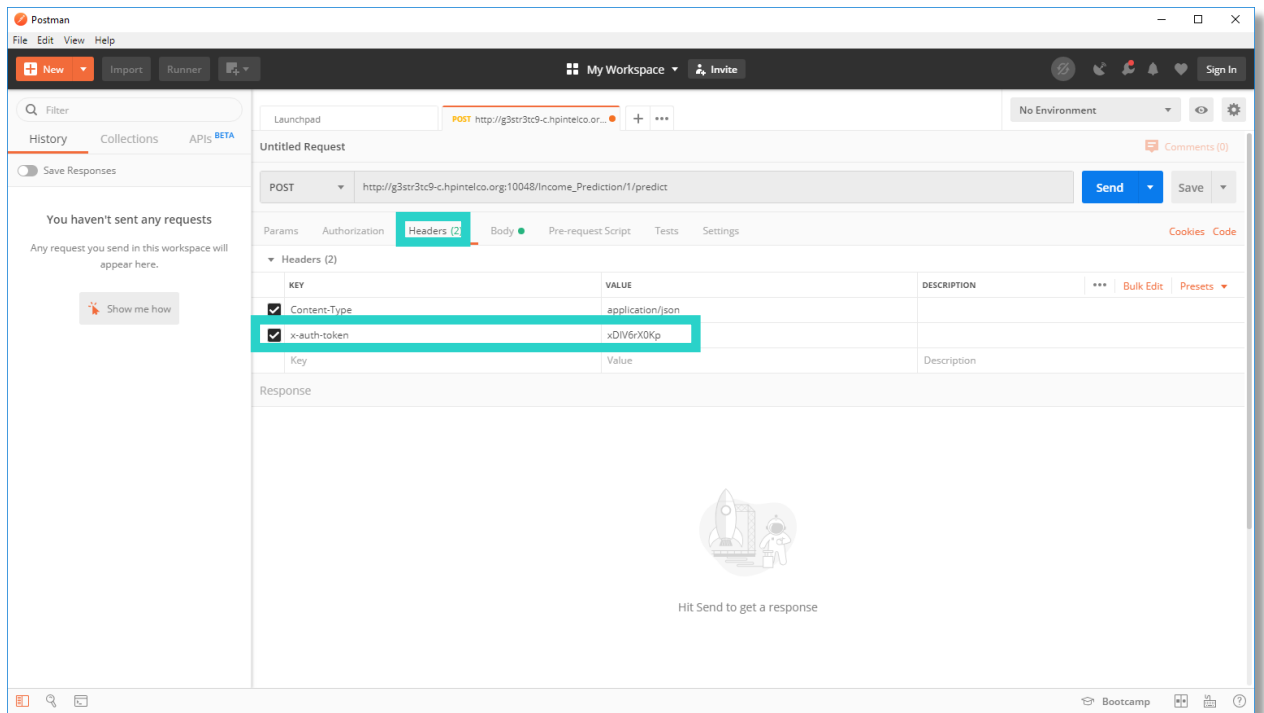
3. Create a new Request



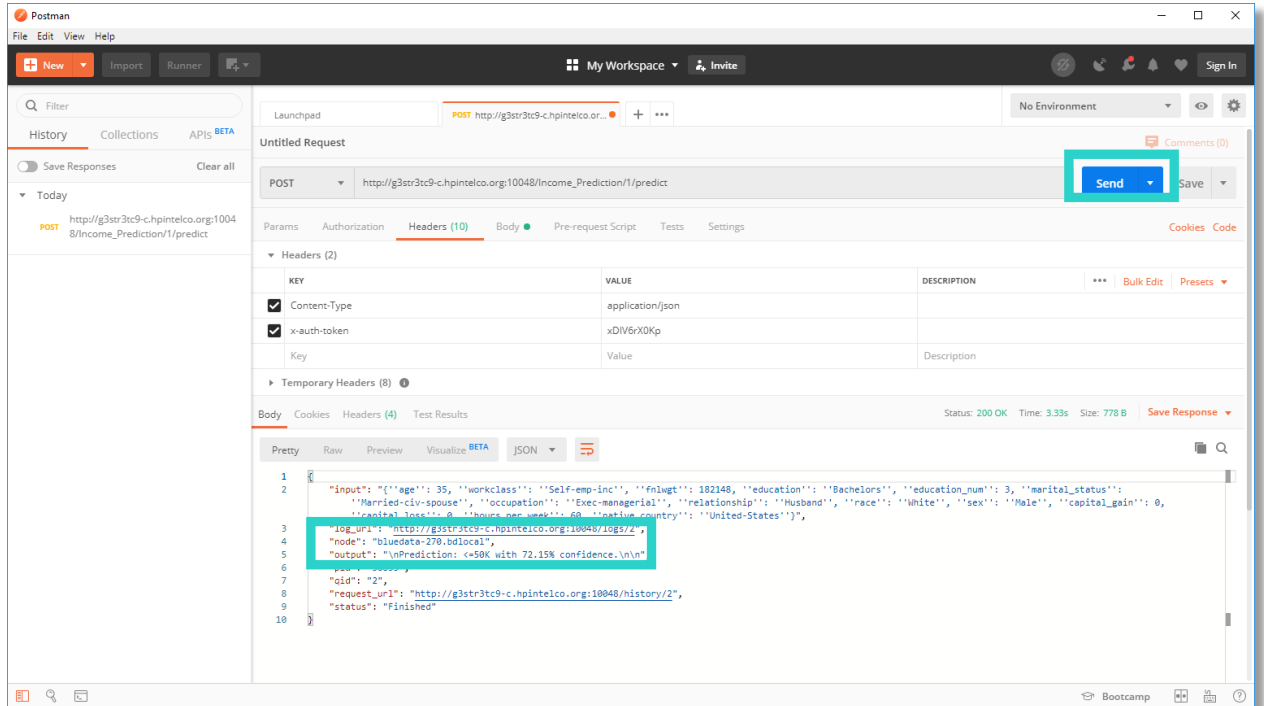
4. Fill in the Body tab to match the image below



5. Fill in the Header tab with your token



6. Send the post request to the cluster and get the income prediction result

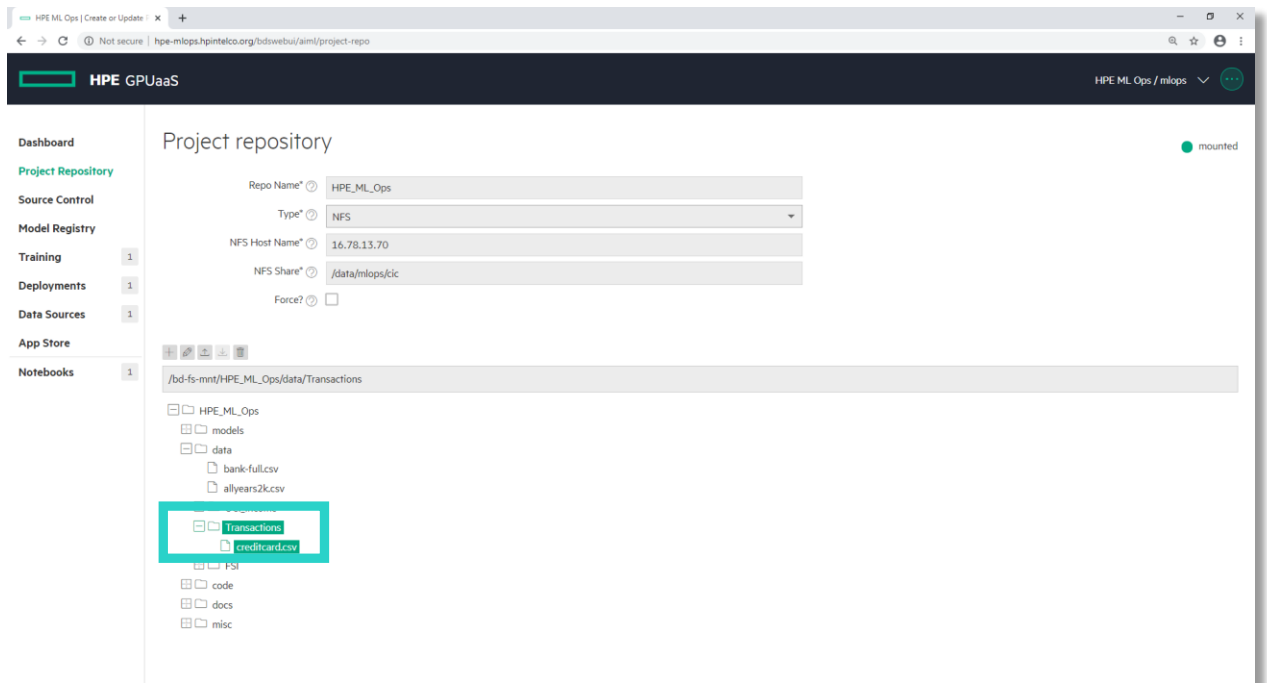


Fraud Detection Demo Scenario

This use case demonstrates using HPE ML Ops and XGBoost to detect credit card fraud.

Data set is in the Project Repository

This dataset contains the credit card transactions that occurred during a period of two days with 492 frauds out of 284,807 transactions. All the variables in the dataset are numerical. The time column contains the seconds elapsed between the first transaction in the dataset.



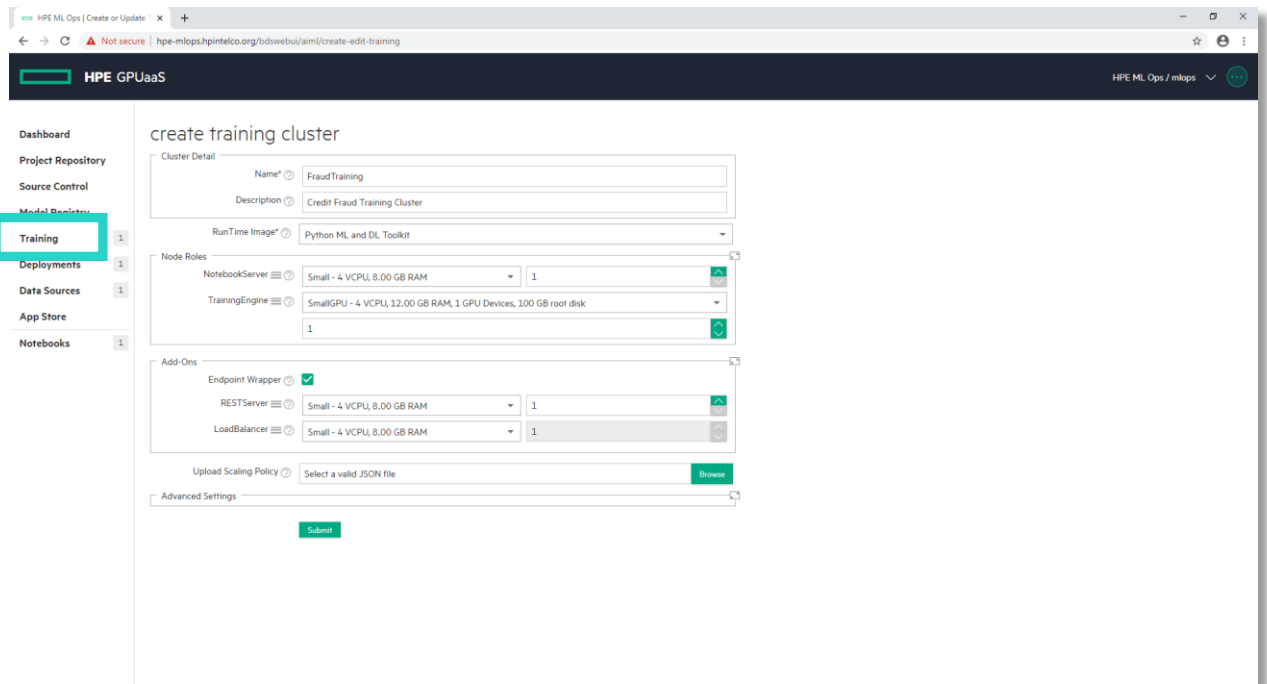
mlops project member (data scientist)

As a project member of ML Ops project, the following are typical tasks a data scientist would perform:

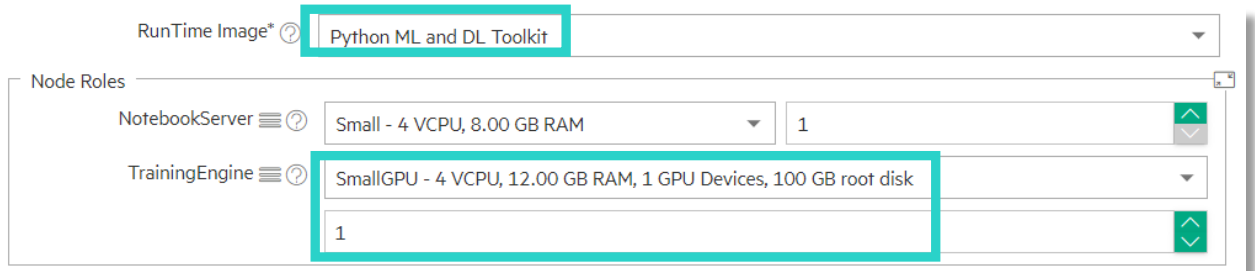
1. Create Training Cluster
2. Create personal Jupyter Notebook
3. Fraud data discover
4. Train XGBoost with CPU
5. Train XGBoost with GPU
6. Register a model
7. Create deployment cluster
8. Fraud prediction with PostMan

Create your HPE ML Ops Training cluster

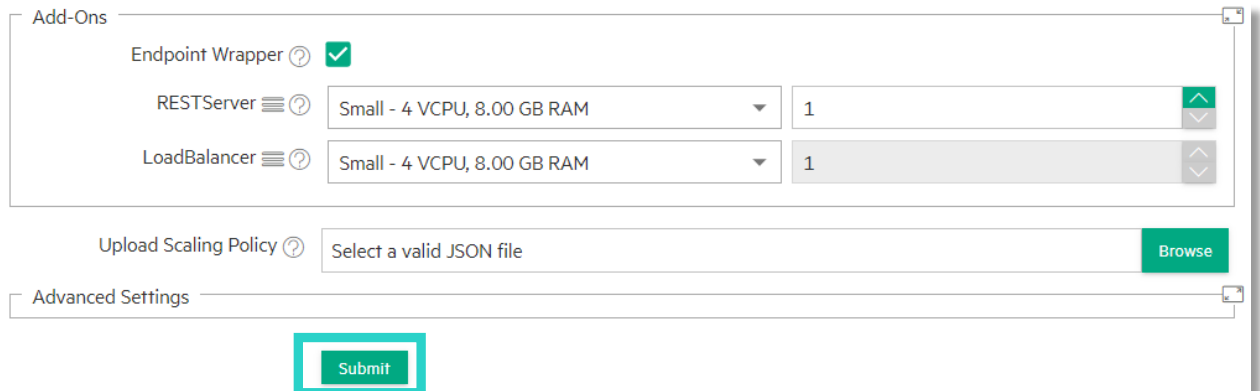
1. Select Create Training



Don't forget to add a Training Engine with GPU flavor and update the RunTime Image



2. Click on Submit



3. Check that the cluster is ready

FraudTraining Python ML and DL Toolkit (i) NotebookServer(1/Small) (i) Created At: Wed Apr 22 2020 12:30:41
 Dependent Distro: Endpoint Wrapper (i) TrainingEngine(1/SmallGPU) (i) Created By: mlops
 RESTServer(1/Small) (i)
 LoadBalancer(1/Small) (i)

4. Open Training Cluster details (Click on FraudTraining)

FraudTraining Python ML and DL Toolkit (i) NotebookServer(1/Small) (i) Created At: Wed Apr 22 2020 12:30:41
 Dependent Distro: Endpoint Wrapper (i) TrainingEngine(1/SmallGPU) (i) Created By: mlops
 RESTServer(1/Small) (i)
 LoadBalancer(1/Small) (i)

5. Show the cluster nodes and services. (NotebookServer / TrainingEngine / ...)

FraudTraining
 [AIML/Training] Credit Fraud Training Cluster

Node(s) Info ActionScript(s) ServiceStatus Cluster Histories

| Name | Distribution | Role | Instance IP |
|----------------------|--------------------------|----------------|-------------|
| bluedata-518.bdlocal | Endpoint Wrapper | RESTServer | 172.18.0.21 |
| bluedata-517.bdlocal | Python ML and DL Toolkit | TrainingEngine | 172.18.0.20 |
| bluedata-516.bdlocal | Python ML and DL Toolkit | NotebookServer | 172.18.0.22 |
| bluedata-519.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.19 |

Services

- API Server : http://g3str3tc9-c.hpintelco.org:10011 [Auth Token]
- SSH : g3str3tc9-c.hpintelco.org -p 10012
- ssh : g3str3tc9-c.hpintelco.org -p 10010
- Standalone Jupyterhub
- ssh : g3str3tc9-c.hpintelco.org -p 10009
- Model serving request balancer stats
- API Server : http://g3str3tc9-c.hpintelco.org:10015 [Auth Token]
- Training API Server : http://g3str3tc9-c.hpintelco.org:10026/train [Auth Token]

6. Check the Services Status of your cluster (Application / System services)

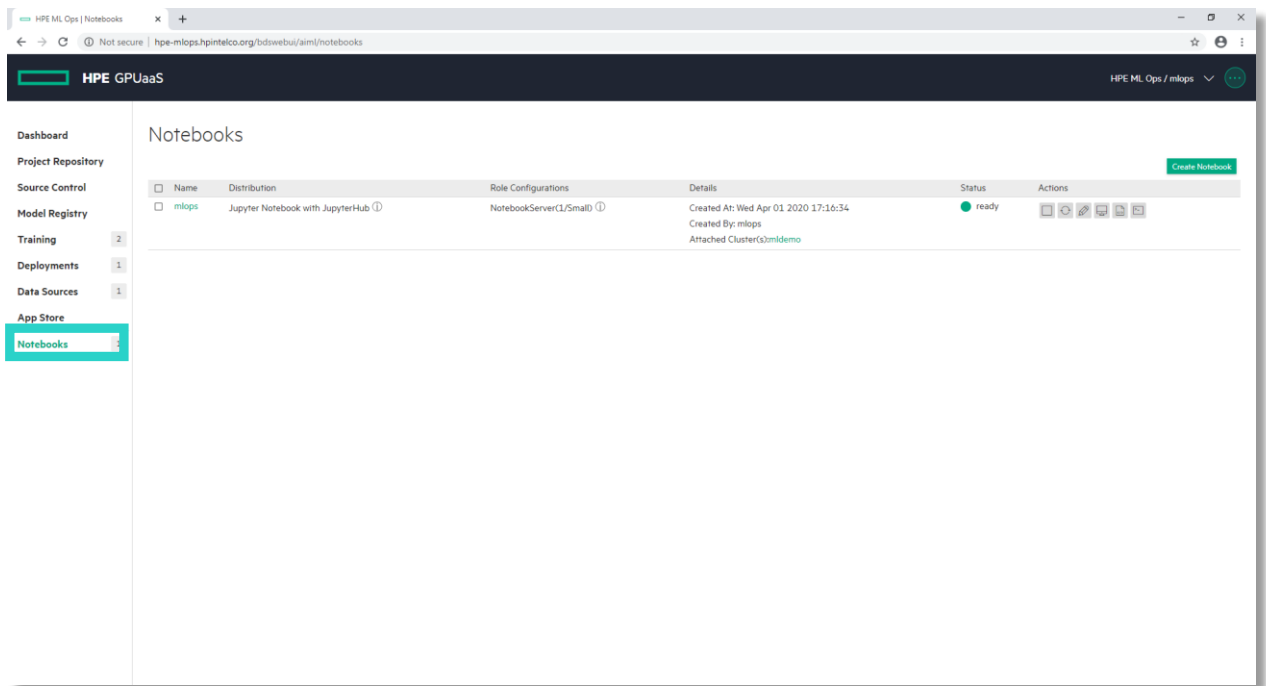
FraudTraining [AIML/Training] Credit Fraud Training Cluster

Node(s) Info ActionScript(s) ServiceStatus Cluster Histories

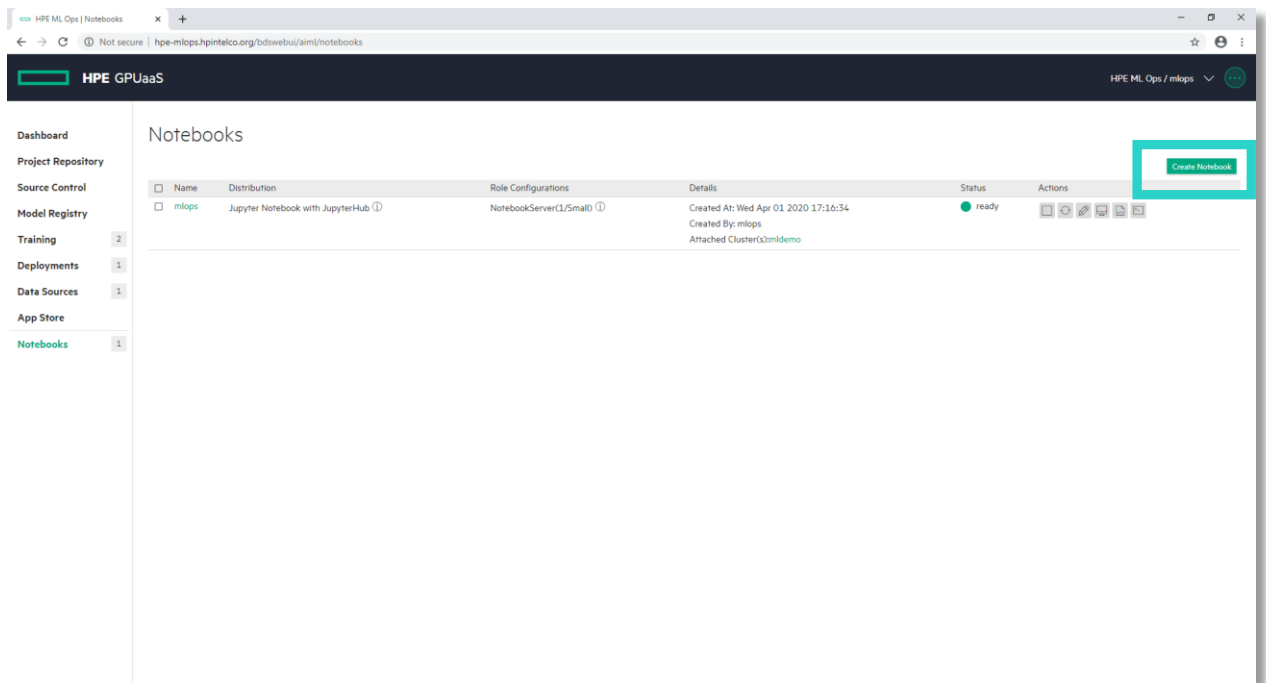
| Name | API Server | Model Serving LoadBalancer | Standalone Jupyterhub | BlueData Agent | User Auth |
|----------------------|------------|----------------------------|-----------------------|----------------|-----------|
| bluedata-519.bdlocal | ● | ● | ● | ● | ● |
| bluedata-518.bdlocal | ● | ● | ● | ● | ● |
| bluedata-517.bdlocal | ● | ● | ● | ● | ● |
| bluedata-516.bdlocal | ● | ● | ● | ● | ● |

Develop/Create Jupyter Notebooks

1. Select Notebooks from the left menu



2. Create a Jupyter Notebook (Click on Create Notebook)



3. Associate the Notebook with your Training cluster

create notebook cluster

Cluster Detail

Name* ? FraudDetection

Description ? HPE Credit Fraud Detection

RunTime Image* ? Jupyter Notebook with JupyterHub

Node Roles

NotebookServer ? SmallGPU 4 VCPU, 12.00 GB RAM, 1 GPU Devices, 100 GB root disk

1

Associate with Training ? FraudTraining

Environments

Advanced Settings

Submit

Don't forget to associate with Training Engine, setup GPU flavor and update the RunTime Image v2

4. Create Notebook Cluster (Click on Submit)

create notebook cluster

Cluster Detail

Name* ? FraudDetection

Description ? HPE Credit Fraud Detection

RunTime Image* ? Jupyter Notebook with JupyterHub

Node Roles

NotebookServer ? SmallGPU - 4 VCPU, 12.00 GB RAM, 1 GPU Devices, 100 GB root disk

1

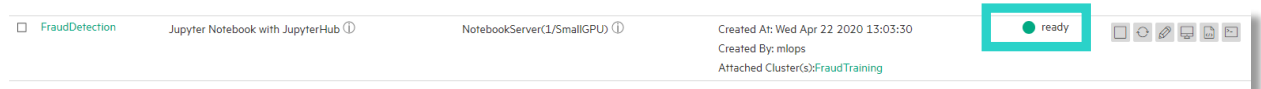
Associate with Training ? FraudTraining

Environments

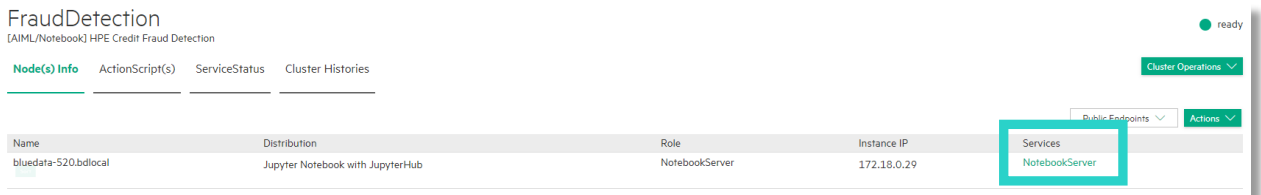
Advanced Settings

Submit

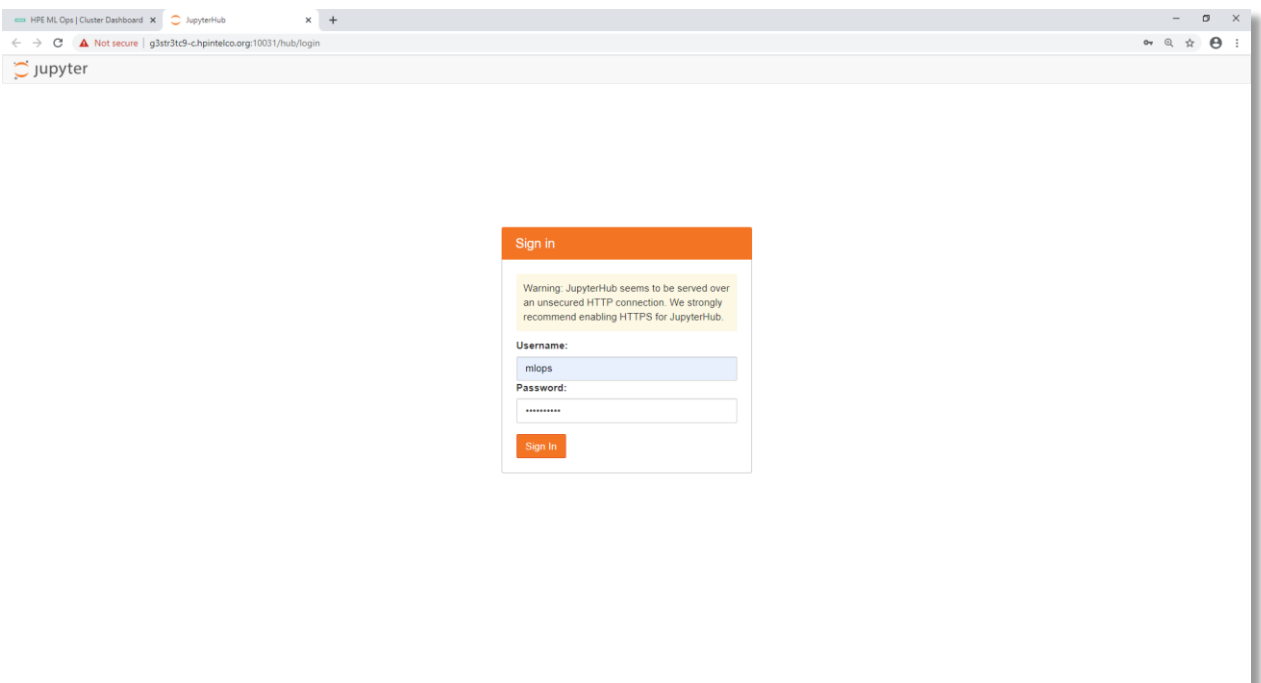
5. Make sure that the status of your notebook is ready



6. Open JupyterHub web ui (Click on NotebookServer)

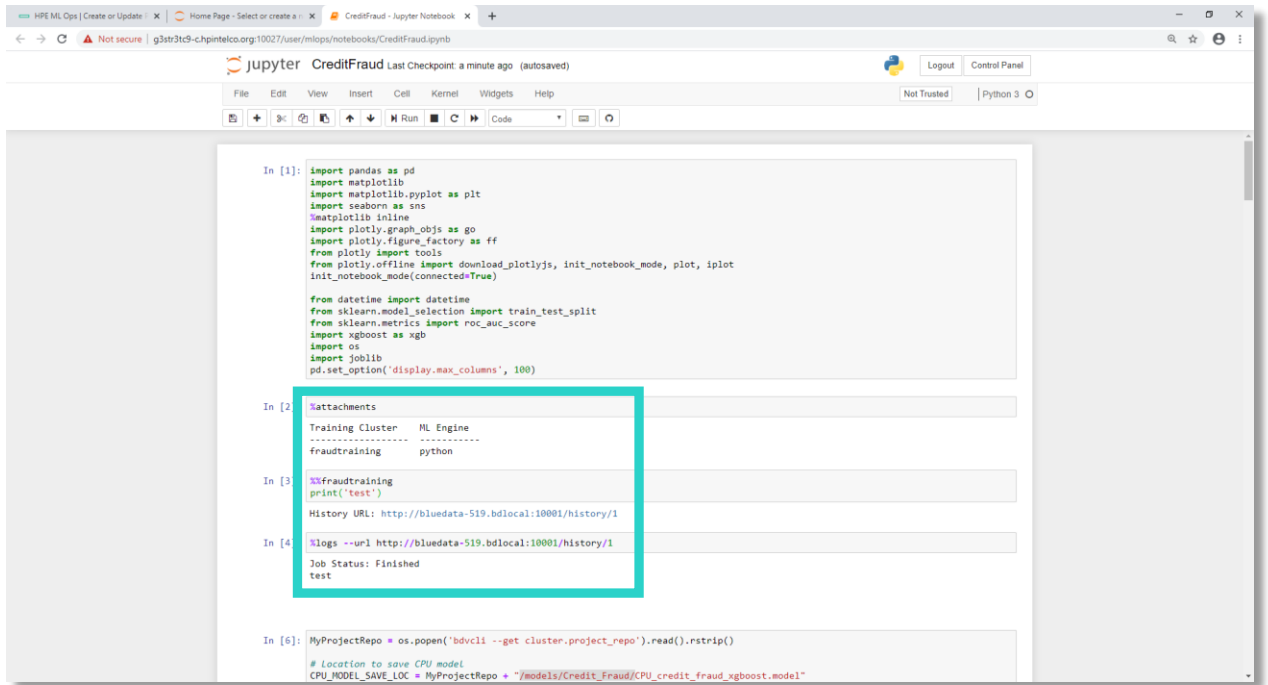


7. Login in to the Jupyter WebUI as mlops user



8. Load CreditFraud.ipynb notebook

Open the `CreditFraud.ipynb` file, and then run each cell individually. This generates model files using CPUs and GPUs. The Notebook contains detailed comments and explanations.



```
In [1]: import pandas as pd
import matplotlib
import matplotlib.pyplot as plt
import seaborn as sns
%matplotlib inline
import plotly.graph_objs as go
import plotly.figure_factory as ff
from plotly import tools
from plotly.offline import download_plotlyjs, init_notebook_mode, plot, iplot
init_notebook_mode(connected=True)

from datetime import datetime
from sklearn.model_selection import train_test_split
from sklearn.metrics import roc_auc_score
import xgboost as xgb
import os
import joblib
pd.set_option('display.max_columns', 100)

In [2]: %attachments
Training Cluster  ML Engine
-----
fraudtraining    python

In [3]: %xgboosttrain
print('test')
History URL: http://bluedata-519.bdlocal:10001/history/1

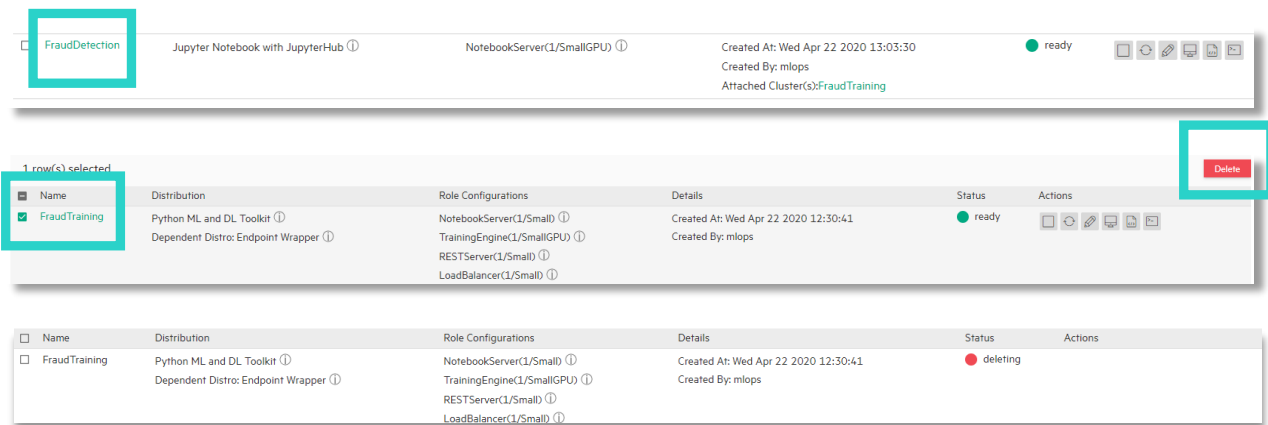
In [4]: %logs --url http://bluedata-519.bdlocal:10001/history/1
Job Status: Finished
test

In [6]: MyProjectRepo = os.popen('bdvc11 --get cluster.project_repo').read().rstrip()
# Location to save CPU model
CPU_MODEL_SAVE_LOC = MyProjectRepo + "/models/Credit_Fraud/CPU_credit_fraud_xgboost.model"
```

(Caution: Update the variable according to your Training Cluster including History urls.)

Once the model is tuned to the optimal parameters, run the cell on a remote Training cluster (where there are potentially more resources to train a model on a larger dataset).

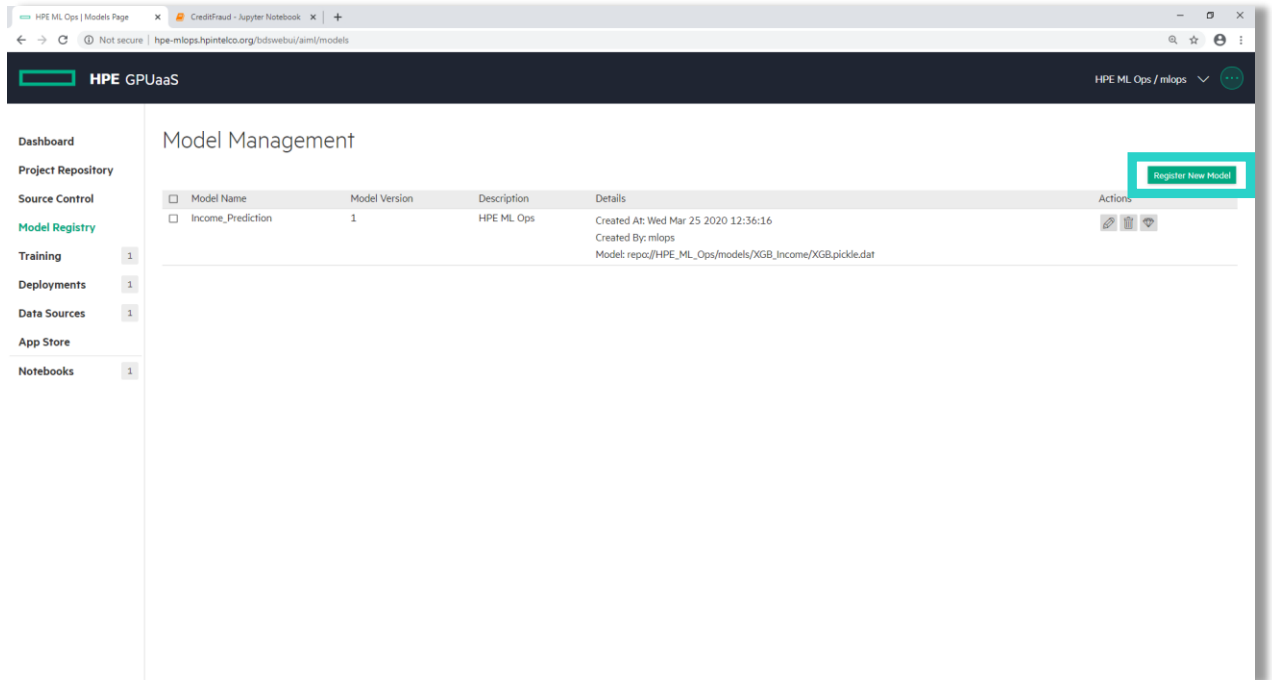
9. Cleanup your Notebook and Training cluster (Needed to release GPU resources)



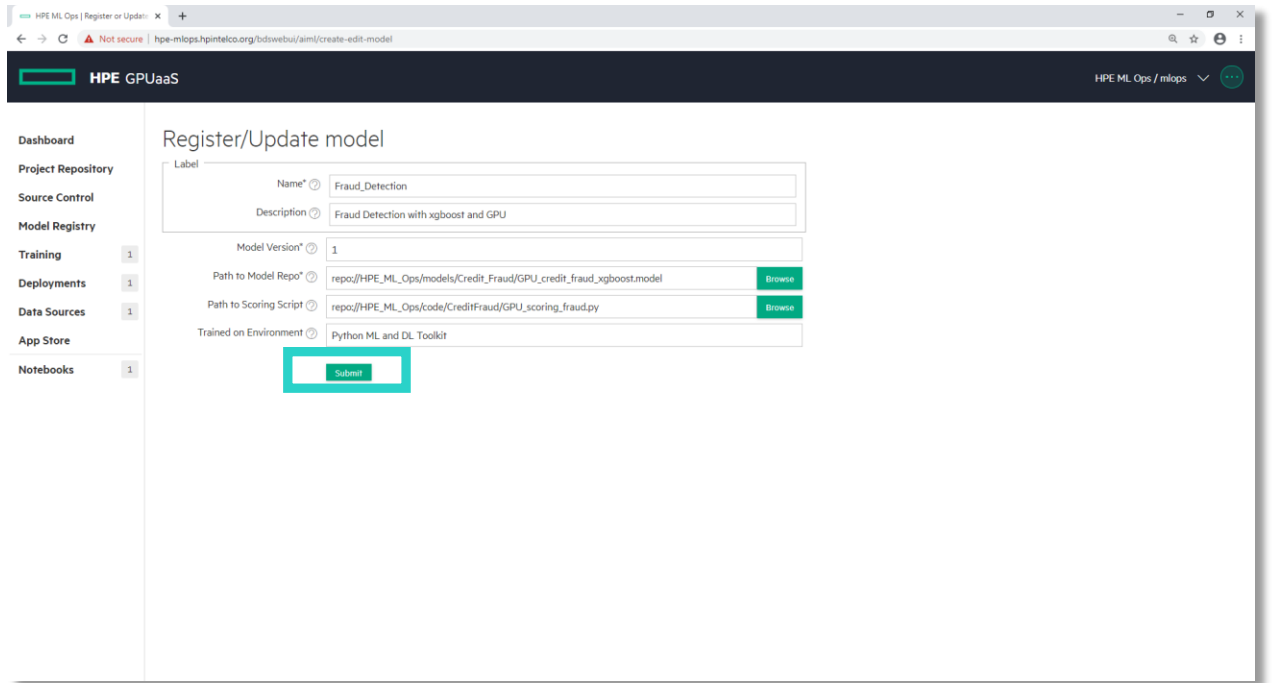
| Name | Distribution | Role Configurations | Details | Status | Actions |
|---|--|--|---|----------|---------|
| <input type="checkbox"/> FraudDetection | Jupyter Notebook with JupyterHub | NotebookServer(1/SmallGPU) | Created At: Wed Apr 22 2020 13:03:30 Created By: mlops Attached Cluster(s): FraudTraining | ready | |
| 1 row(s) selected | | | | | |
| <input checked="" type="checkbox"/> FraudTraining | Python ML and DL Toolkit Dependent Distro: Endpoint Wrapper | NotebookServer(1/Small) TrainingEngine(1/SmallGPU) RETServer(1/Small) LoadBalancer(1/Small) | Created At: Wed Apr 22 2020 12:30:41 Created By: mlops | ready | |
| <input type="checkbox"/> FraudTraining | Python ML and DL Toolkit Dependent Distro: Endpoint Wrapper | NotebookServer(1/Small) TrainingEngine(1/SmallGPU) RETServer(1/Small) LoadBalancer(1/Small) | Created At: Wed Apr 22 2020 12:30:41 Created By: mlops | deleting | |

Register and Deploy the Model

1. Select Model Registry from the left menu and click on Register New Model




2. Register New Model using the information displayed below. (GPU / CPU option)





3. GPU option


Register/Update model


Label


Name*  Fraud_Detection

Description  Fraud Detection with xgboost and GPU

Model Version*  1

Path to Model Repo*  repo://HPE_ML_Ops/models/Credit_Fraud/GPU_credit_fraud_xgboost.model [Browse](#)

Path to Scoring Script  repo://HPE_ML_Ops/code/CreditFraud/GPU_scoring_fraud.py [Browse](#)


Trained on Environment  Python ML and DL Toolkit


[Submit](#)


4. CPU option


Register/Update model


Label


Name*  Fraud_Detection_CPU

Description  Fraud Detection with xgboost and CPU

Model Version*  1

Path to Model Repo*  repo://HPE_ML_Ops/models/Credit_Fraud/CPU_credit_fraud_xgboost.model [Browse](#)

Path to Scoring Script  repo://HPE_ML_Ops/code/CreditFraud/CPU_scoring_fraud.py [Browse](#)










Trained on Environment  Python ML and DL Toolkit

[Submit](#)

5. Deploy your registered model (Click on Deploy)

Model Management

[Register New Model](#)

| <input type="checkbox"/> | Model Name | Model Version | Description | Details | Actions |
|--------------------------|---------------------|---------------|--------------------------------------|--|---|
| <input type="checkbox"/> | Fraud_Detection | 1 | Fraud Detection with xgboost and GPU | Created At: Wed Apr 22 2020 15:29:03 Created By: mlops Model: repo://HPE_ML_Ops/models/Credit_Fraud/GPU_credit_fraud_xgboost.model |    |
| <input type="checkbox"/> | Fraud_Detection_CPU | 1 | Fraud Detection with xgboost and CPU | Created At: Wed Apr 22 2020 15:21:13 Created By: mlops Model: repo://HPE_ML_Ops/models/Credit_Fraud/CPU_credit_fraud_xgboost.model |    |
| <input type="checkbox"/> | Income_Prediction | 1 | HPE ML Ops | Created At: Wed Mar 25 2020 12:36:16 Created By: mlops Model: repo://HPE_ML_Ops/models/XGB_Income/XGB.pickle.dat |    |

6. Create a Deployment Cluster for Credit Fraud Prediction (Click on Submit)

create deployment cluster

Cluster Detail

Name*

Description

Select Model

RunTime Image*

Node Roles

InferenceEngine

Add-Ons

Endpoint Wrapper

RESTServer

LoadBalancer

Upload Scaling Policy

Advanced Settings

7. Credit Fraud Prediction Cluster is now Ready

Deployments

| Name | Distribution | Role Configurations | Details | Status | Actions |
|---|--|---|---|--|--|
| <input type="checkbox"/> Income_Prediction | Python ML/DL Toolkit Dependent Distro: Endpoint Wrapper | InferenceEngine(1/Small) RESTServer(1/Small) LoadBalancer(1/Small) | Created At: Wed Mar 25 2020 12:37:53 Created By: mllops Attached Model(s): Income_Prediction, 1 | ● ready | <input type="button" value="Refresh"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Download"/> <input type="button" value="Upload"/> |
| <input checked="" type="checkbox"/> CreditFraud | Python ML/DL Toolkit Dependent Distro: Endpoint Wrapper | InferenceEngine(1/SmallGPU) RESTServer(1/Small) LoadBalancer(1/Small) | Created At: Wed Apr 22 2020 15:33:19 Created By: mllops Attached Model(s): Fraud_Detection, 1 | ● ready | <input type="button" value="Refresh"/> <input type="button" value="Edit"/> <input type="button" value="Delete"/> <input type="button" value="Download"/> <input type="button" value="Upload"/> |

CreditFraud

[AIML/Deployment] Credit Fraud Detection

Node(s) Info | ActionScript(s) | ServiceStatus | Cluster Histories

● ready

Cluster Operations

Public Endpoints Actions

| Name | Distribution | Role | Instance IP | Services |
|----------------------|----------------------|-----------------|-------------|---|
| bluedata-522.bdlocal | Endpoint Wrapper | RESTServer | 172.18.0.20 | API Server : http://g3str3tc9-c.hpintelco.org:10009 [Auth Token] SSH : g3str3tc9-c.hpintelco.org -p 10010 |
| bluedata-521.bdlocal | Python ML/DL Toolkit | InferenceEngine | 172.18.0.21 | ssh : g3str3tc9-c.hpintelco.org -p 10008 |
| bluedata-523.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.19 | Model serving request balancer stats API Server : http://g3str3tc9-c.hpintelco.org:10012 [Auth Token] Model Serving LoadBalancer : http://g3str3tc9-c.hpintelco.org:10013/<model_name>/<model_version>/predict [Auth Token] |

8. Generate your Prediction Request

CreditFraud [AIML/Deployment] Credit Fraud Detection ready

Node(s) Info ActionScript(s) ServiceStatus Cluster Histories Cluster Operations

| Name | Distribution | Role | Instance IP | Services |
|----------------------|----------------------|-----------------|-------------|---|
| bluedata-522.bdlocal | Endpoint Wrapper | RESTServer | 172.18.0.20 | API Server : http://g3str3tc9-c.hpintelco.org:10009 [Auth Token] SSH : g3str3tc9-c.hpintelco.org -p 10010 |
| bluedata-521.bdlocal | Python ML/DL Toolkit | InferenceEngine | 172.18.0.21 | sshd : g3str3tc9-c.hpintelco.org -p 10008 |
| bluedata-523.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.19 | Model serving request balancer stats API Server : http://g3str3tc9-c.hpintelco.org:10012 [Auth Token] loadBalancer : http://g3str3tc9-c.hpintelco.org:10013/<model_name>/<model_version>/predict [Auth Token] |

Copy the link and check update the port + model version

http://g3str3tc9-c.hpintelco.org:10013/Fraud_Detection/1/predict
[Auth Token] Click to copy the token

Non-Fraudulent Request

```
{
  "use_scoring" : true,
  "scoring_args" : {
    "transaction": "0,-1.359807134,-0.072781173,2.536346738,1.378155224,-
0.33832077,0.462387778,0.239598554,0.098697901,0.36378697,0.090794172,-0.551599533,-0.617800856,-0.991389847,-
0.311169354,1.468176972,-0.470400525,0.207971242,0.02579058,0.40399296,0.251412098,-0.018306778,0.277837576,-
0.11047391,0.066928075,0.128539358,-0.189114844,0.133558377,-0.021053053,149.62,0",
    "execCount": 1
  }
}
```

Fraudulent Request

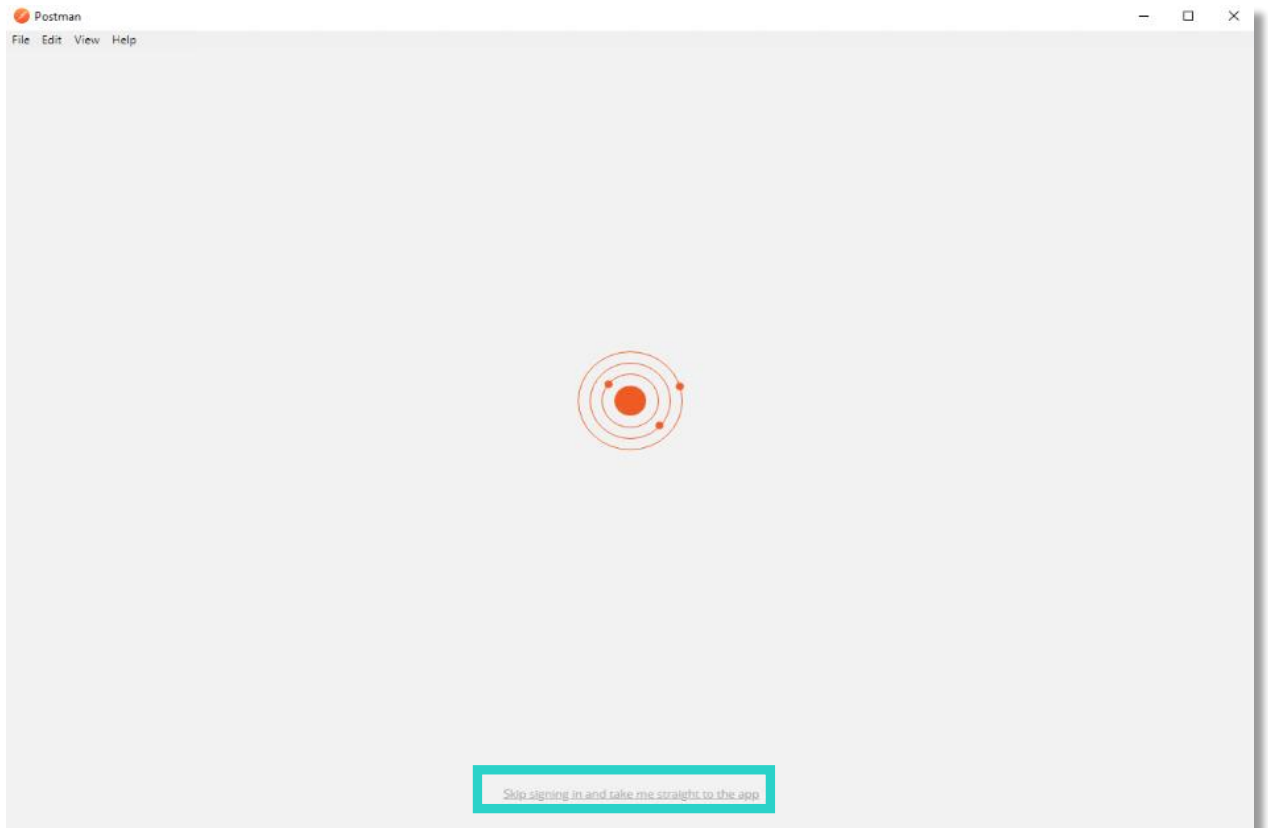
```
{
  "use_scoring" : true,
  "scoring_args" : {
    "transaction": "406,-2.3122265423263,1.95199201064158,-1.60985073229769,3.9979055875468,-0.522187864667764,-
1.42654531920595,-2.53738730624579,1.39165724829804,-2.77008927719433,-2.77227214465915,3.20203320709635,-
2.89990738849473,-0.595221881324605,-4.28925378244217,0.389724120274487,-1.14074717980657,-2.83005567450437,-
0.0168224681808257,0.416955705037907,0.126910559061474,0.517232370861764,-0.0350493686052974,-
0.465211076182388,0.320198198514526,0.0445191674731724,0.177839798284401,0.261145002567677,-
0.143275874698919,0,0",
    "execCount": 1
  }
}
```

Run the Fraud Detection prediction using Postman

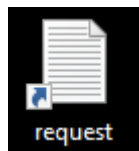


This link will start Postman!

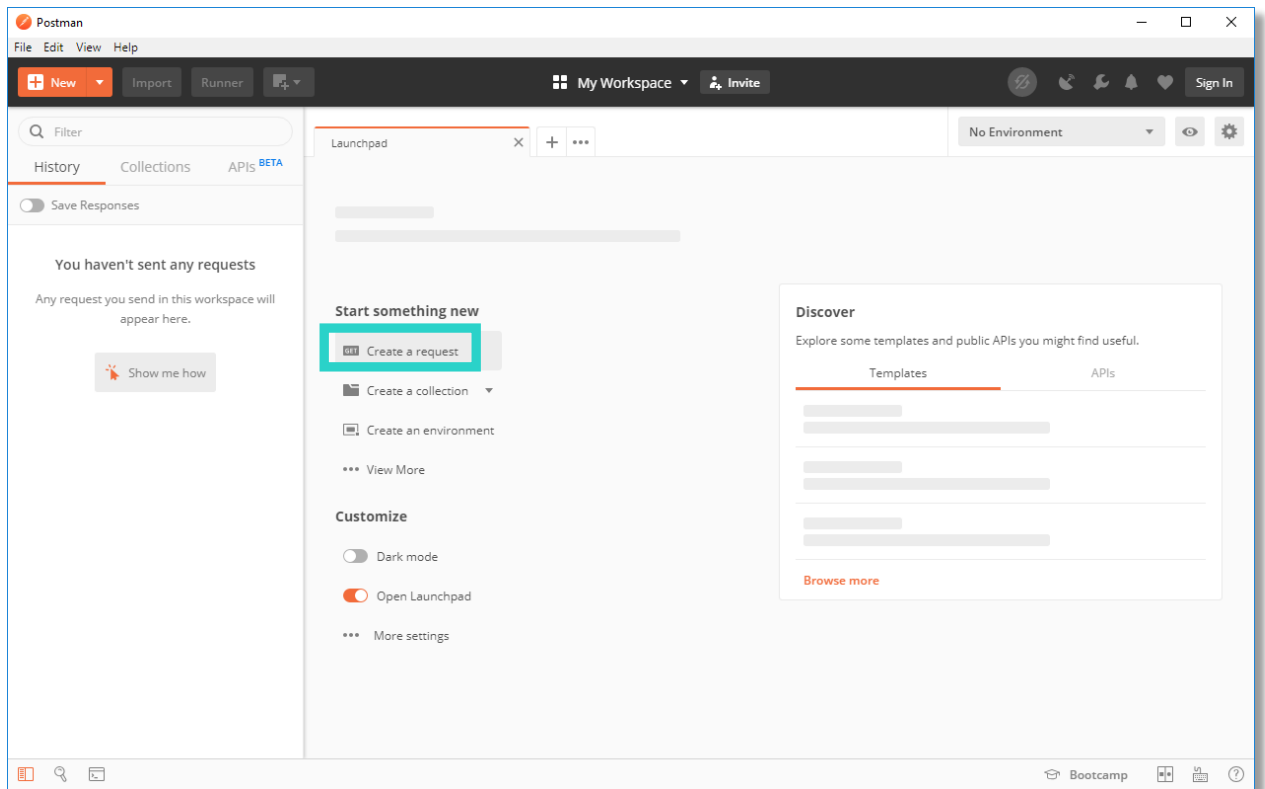
9. Select Skip signing in to go directly to the app



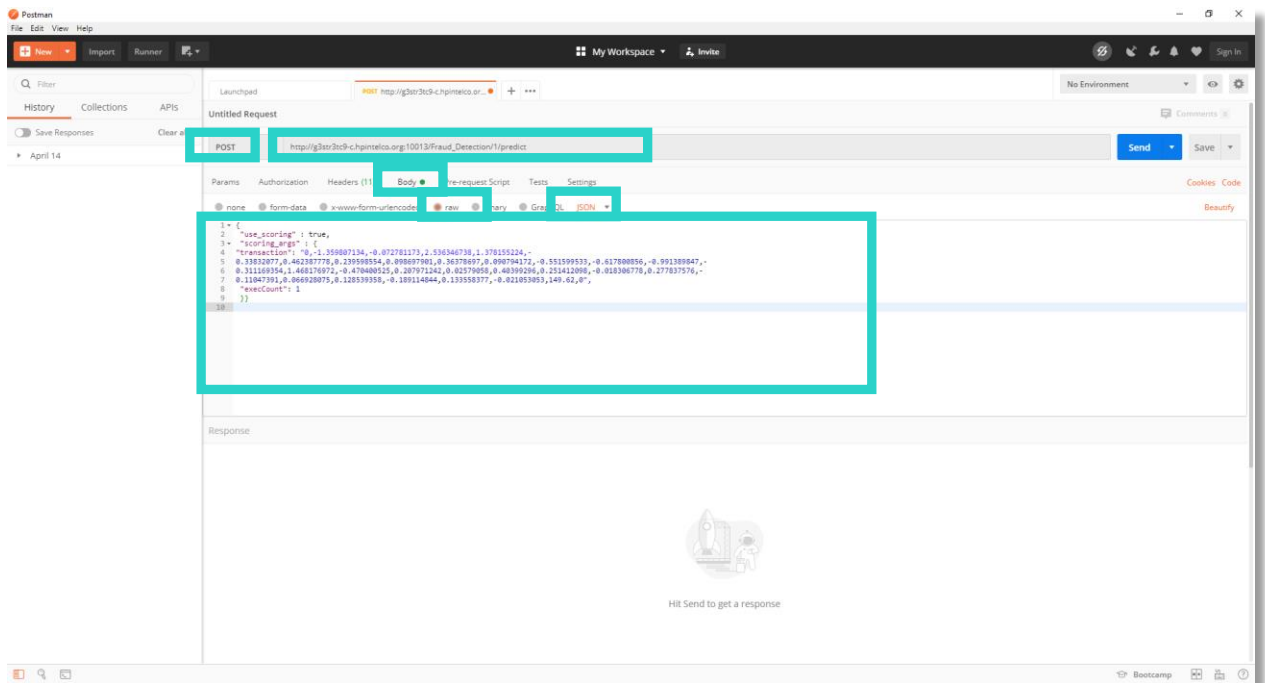
10. Copy the Json request in Postman body tab: Click on below icon



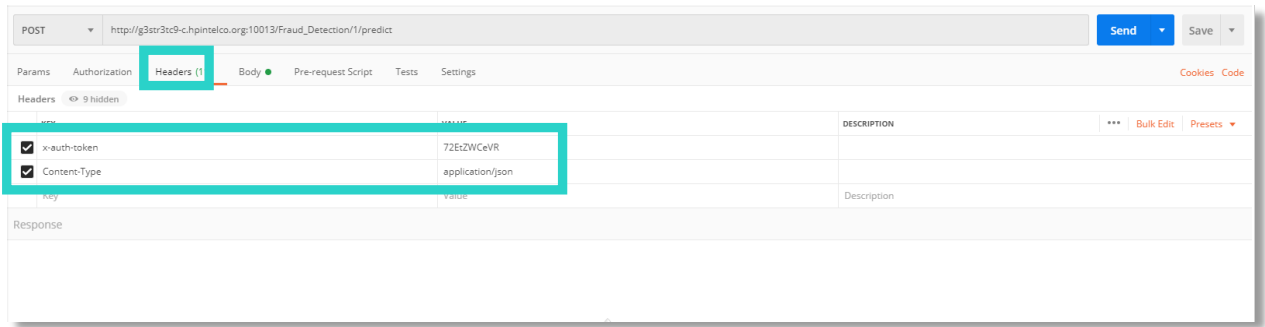
11. Create a new Request



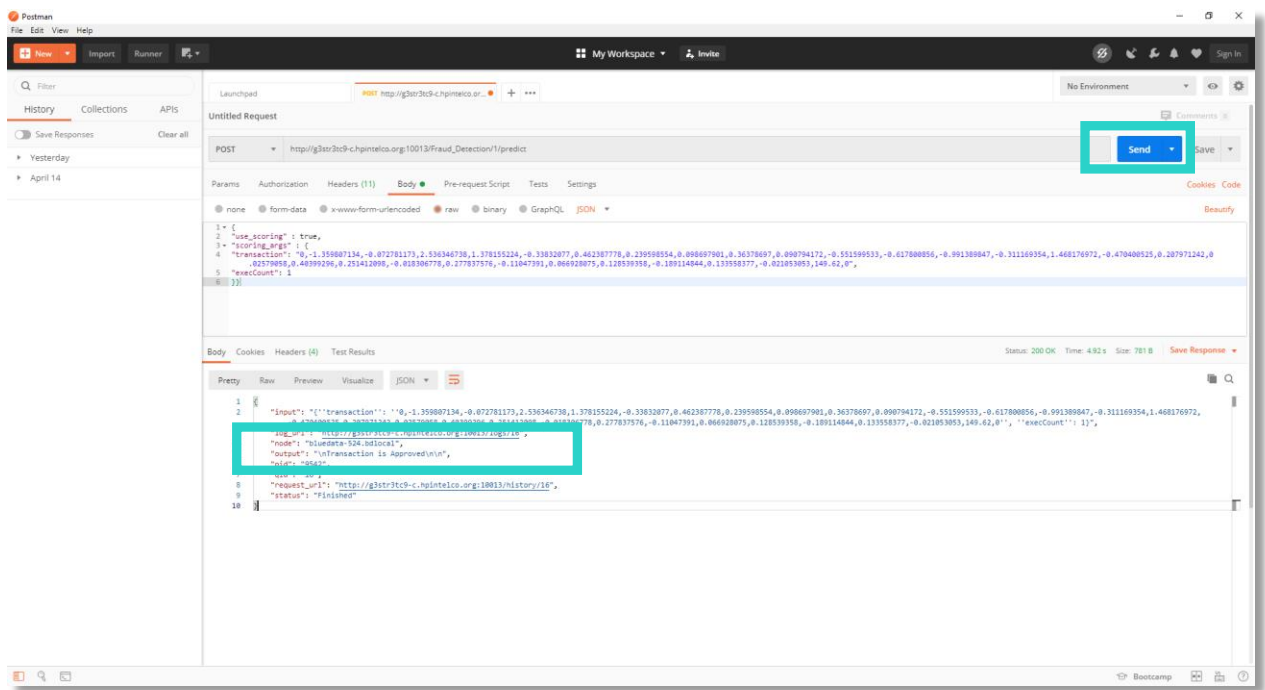
12. Fill in the Body tab to match the image below



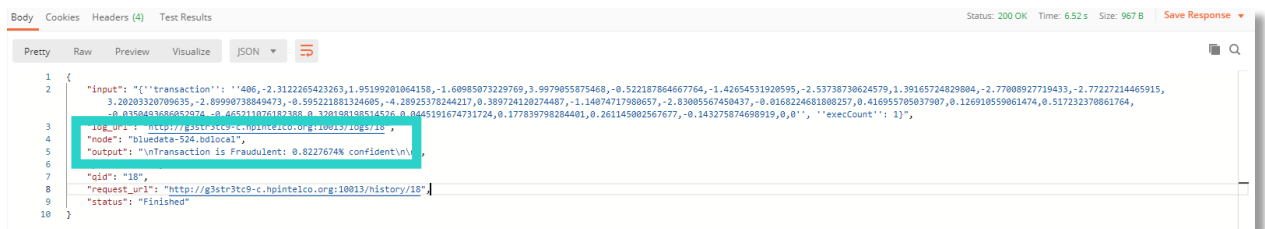
13. Fill in the Header tab with your token



14. Send the post request to the cluster and get the Credit Fraud prediction result



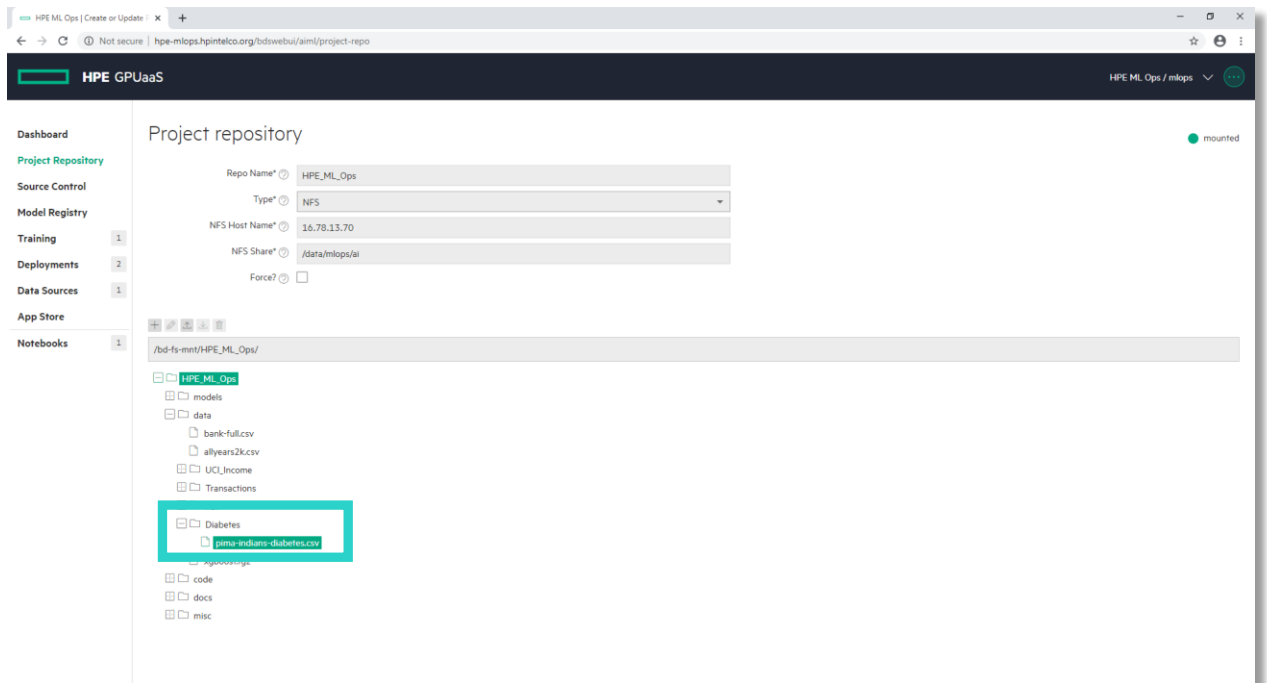
15. Fraudulent Request



Diabetes Prediction Demo Scenario

This tutorial generates a model that classifies whether a person has diabetes or not taking into account a dataset that contains data from female patients who were at least 21 years old and of Pima Indian heritage...

Data set is in the Project Repository

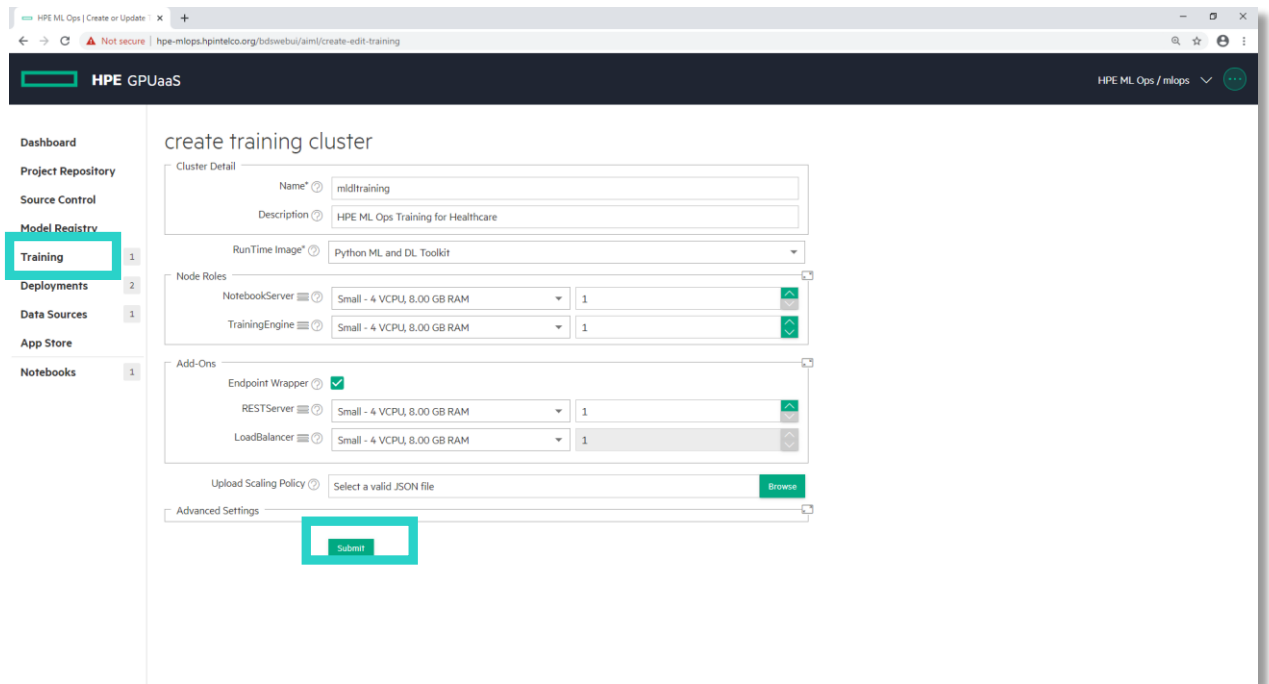


The features (columns) in this spreadsheet that are used to train the model are:

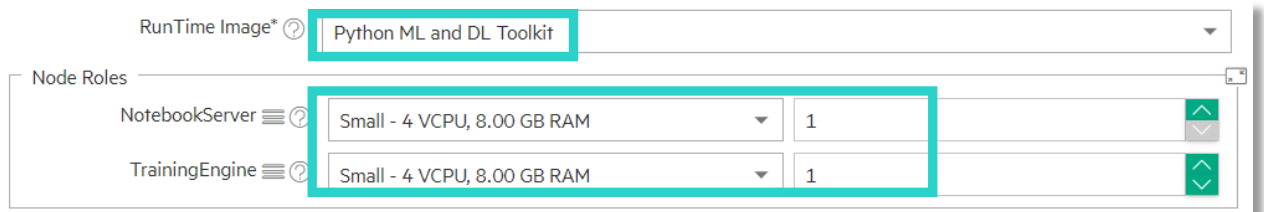
- Pregnancies
- Glucose — The blood plasma glucose concentration after a 2 hour oral glucose tolerance test.
- BloodPressure — Diastolic blood pressure (mm/HG).
- SkinThickness — Skinfold thickness of the triceps (mm).
- Insulin—2 hour serum insulin (μ U/ml).
- BMI—Body mass index (kg/m squared)
- DiabetesPedigreeFunction: Function that determines the risk of type 2 diabetes based on family history.
- Age
- Outcome—whether the person is diagnosed with type 2 diabetes (1 = yes, 0 = no).

Create your HPE ML Ops Training cluster

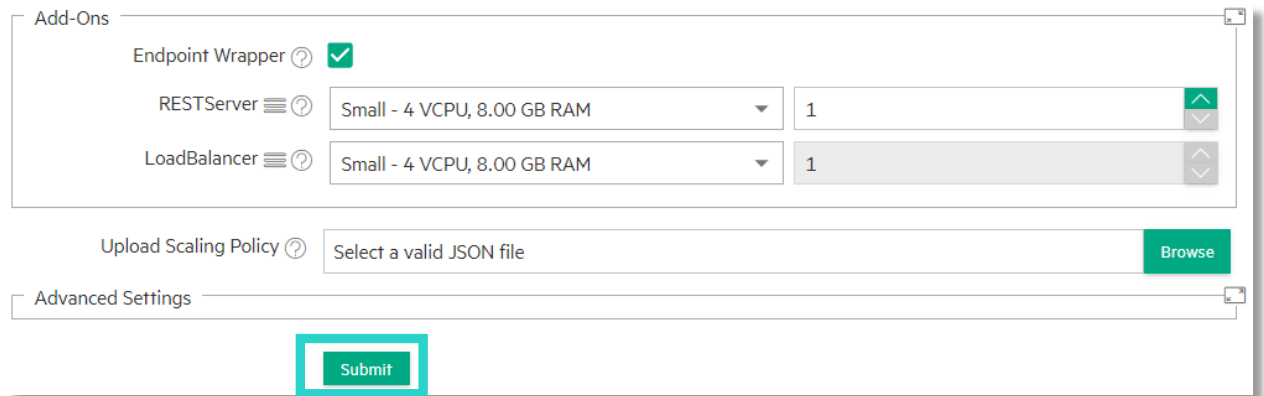
1. Select Create training cluster



Don't forget to add a Training Engine with CPU flavor and update the RunTime Image



2. Click on Submit



3. Check that the cluster is ready

| Name | Distribution | Role Configurations | Details | Status | Actions |
|---------------------------------------|--|--|---|--|---|
| <input type="checkbox"/> mldltraining | Python ML and DL Toolkit ⓘ Dependent Distro: Endpoint Wrapper ⓘ | NotebookServer(1/Small) ⓘ TrainingEngine(1/Small) ⓘ RETSer(1/Small) ⓘ LoadBalancer(1/Small) ⓘ | Created At: Tue May 19 2020 10:10:22 Created By: mlops | ● ready |      |

4. Open Training Cluster details (Click on mldltraining)

| Name | Distribution | Role Configurations | Details | Status | Actions |
|--|--|--|---|--|---|
| <input checked="" type="checkbox"/> mldltraining | Python ML and DL Toolkit ⓘ Dependent Distro: Endpoint Wrapper ⓘ | NotebookServer(1/Small) ⓘ TrainingEngine(1/Small) ⓘ RETSer(1/Small) ⓘ LoadBalancer(1/Small) ⓘ | Created At: Tue May 19 2020 10:10:22 Created By: mlops | ● ready |      |

5. Show the cluster nodes and services. (NotebookServer / TrainingEngine / ...)

mldltraining
[AIML/Training] HPE ML Ops Training for Healthcare ● ready

Node(s) Info | ActionScript(s) | ServiceStatus | Cluster Histories Cluster Operations ▾

Public Endpoints ▾ | Actions ▾

| Name | Distribution | Role | Instance IP | Services |
|----------------------|--------------------------|----------------|-------------|---|
| bluedata-735.bdlocal | Python ML and DL Toolkit | TrainingEngine | 172.18.0.34 | ssh : g3str3tc9-c.hpintelco.org -p 10039 |
| bluedata-736.bdlocal | Endpoint Wrapper | RETSer | 172.18.0.35 | API Server : http://g3str3tc9-c.hpintelco.org:10040 [Auth Token] SSH : g3str3tc9-c.hpintelco.org -p 10041 |
| bluedata-734.bdlocal | Python ML and DL Toolkit | NotebookServer | 172.18.0.37 | Standalone Jupyterhub ssh : g3str3tc9-c.hpintelco.org -p 10038 |
| bluedata-737.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.36 | Model serving request balancer stats API Server : http://g3str3tc9-c.hpintelco.org:10043 [Auth Token] Training API Server : http://g3str3tc9-c.hpintelco.org:10044/train [Auth Token] |

6. Check the Services Status of your cluster (Application / System services)

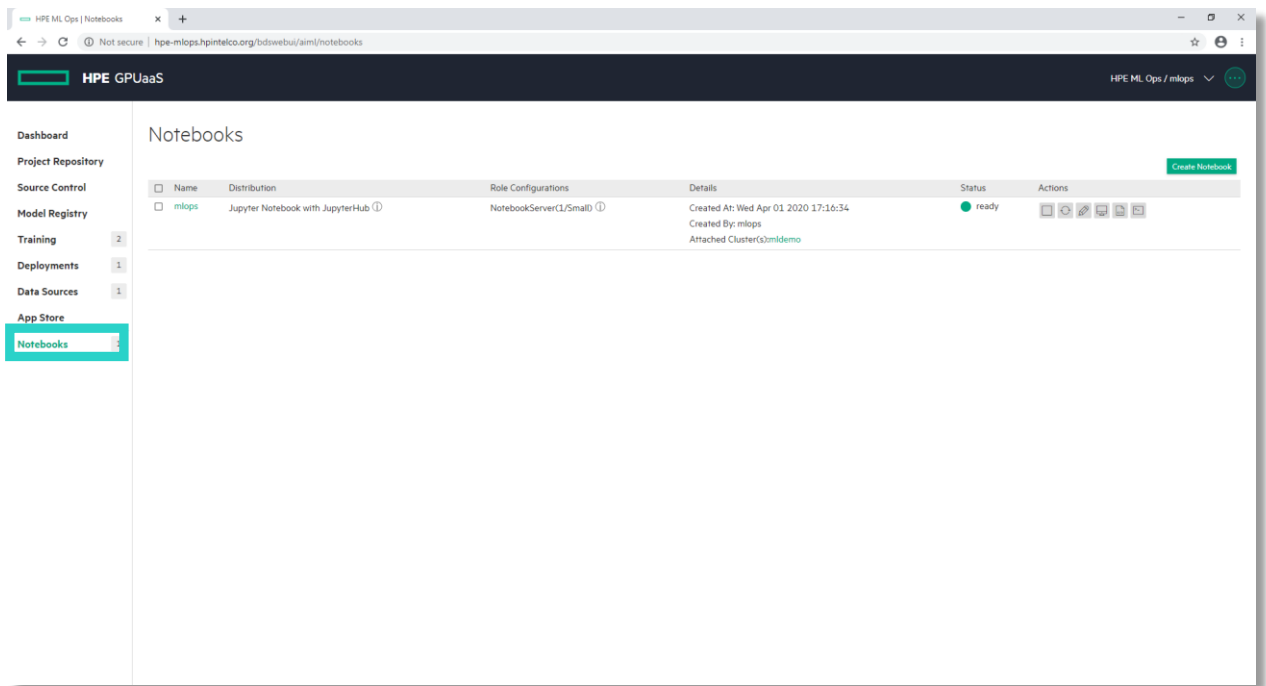
mldltraining
[AIML/Training] HPE ML Ops Training for Healthcare ● ready

Node(s) Info | ActionScript(s) | **ServiceStatus** | Cluster Histories Cluster Operations ▾

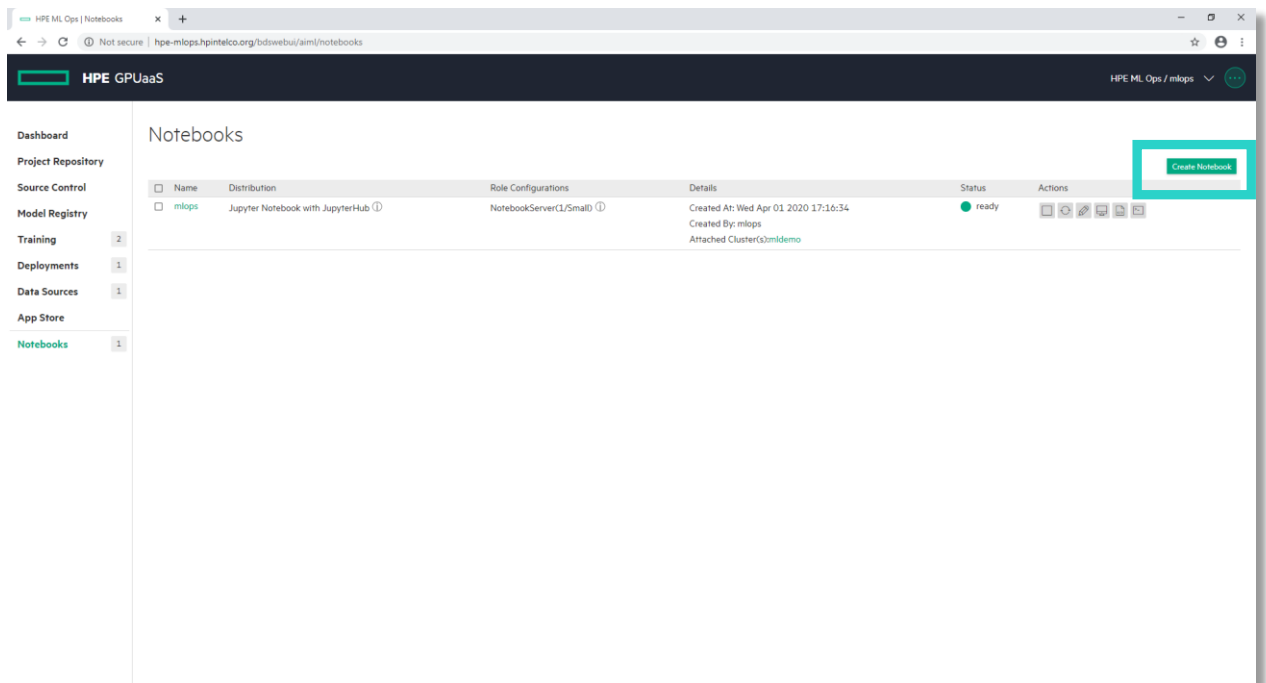
| Name | API Server | Model Serving LoadBalancer | Standalone Jupyterhub | BlueData Agent | User Auth |
|----------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|--------------------------------------|
| bluedata-737.bdlocal | ● | ● | ● | ● | ● |
| bluedata-736.bdlocal | ● | ● | ● | ● | ● |
| bluedata-735.bdlocal | ● | ● | ● | ● | ● |
| bluedata-734.bdlocal | ● | ● | ● | ● | ● |

Develop/Create Jupyter Notebooks

1. Select Notebooks from the left menu



2. Create a Jupyter Notebook (Click on Create Notebook)



3. Associate the Notebook with your Training cluster

create notebook cluster

Cluster Detail

Name* Notebook

Description Healthcare

RunTime Image* Jupyter Notebook with JupyterHub v2

Node Roles

NotebookServer Small - 4 VCPU, 8.00 GB RAM 1

Associate with Training Environments mldtraining

Advanced Settings

Submit

Don't forget to associate with Training Engine, setup CPU flavor and update the RunTime Image v2

4. Create Notebook Cluster (Click on Submit)

create notebook cluster

Cluster Detail

Name* Notebook

Description Healthcare

RunTime Image* Jupyter Notebook with JupyterHub v2

Node Roles

NotebookServer Small - 4 VCPU, 8.00 GB RAM 1

Associate with Training Environments mldtraining

Advanced Settings

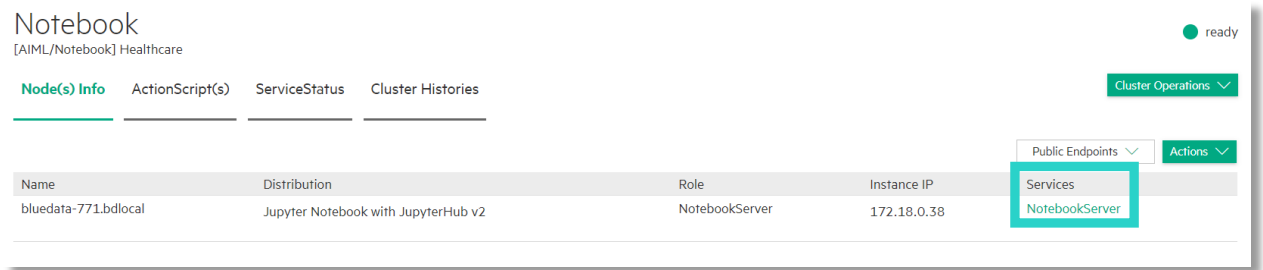
Submit

5. Make sure that the status of your notebook is ready

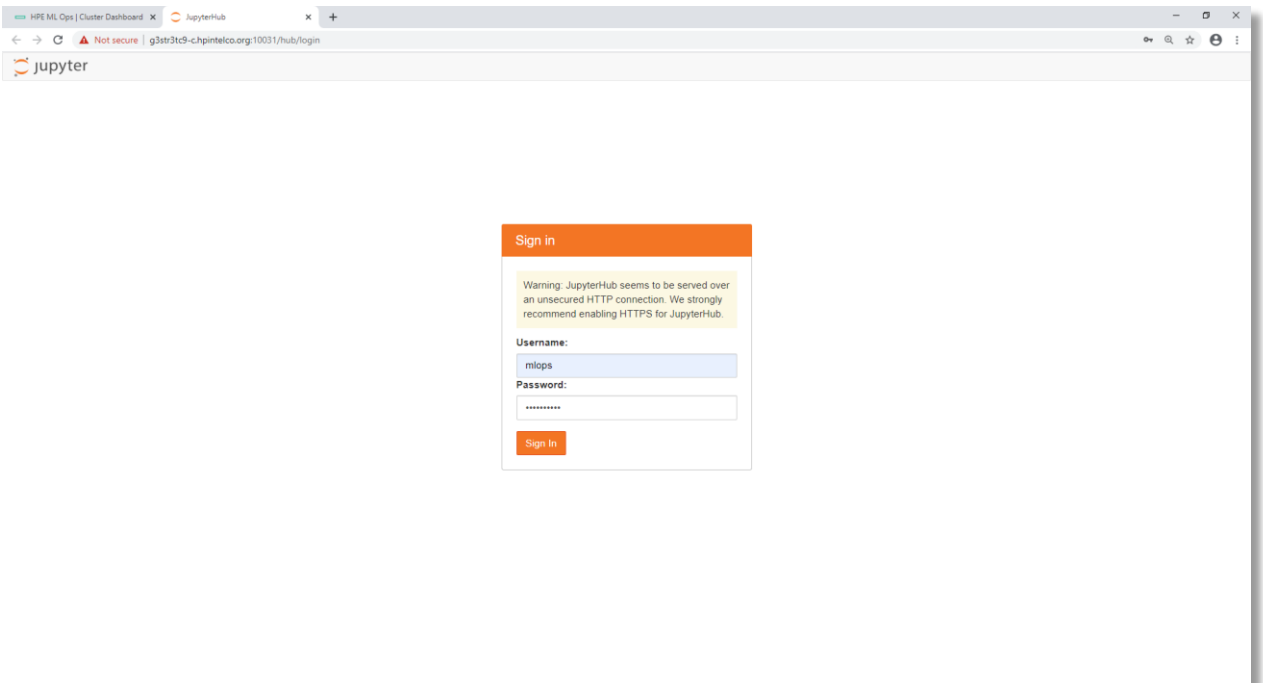
□ Notebook Jupyter Notebook with JupyterHub v2 NotebookServer(1/Small) Created At: Tue May 19 2020 10:25:17 Created By: mllops Attached Cluster(s):mldtraining

● ready

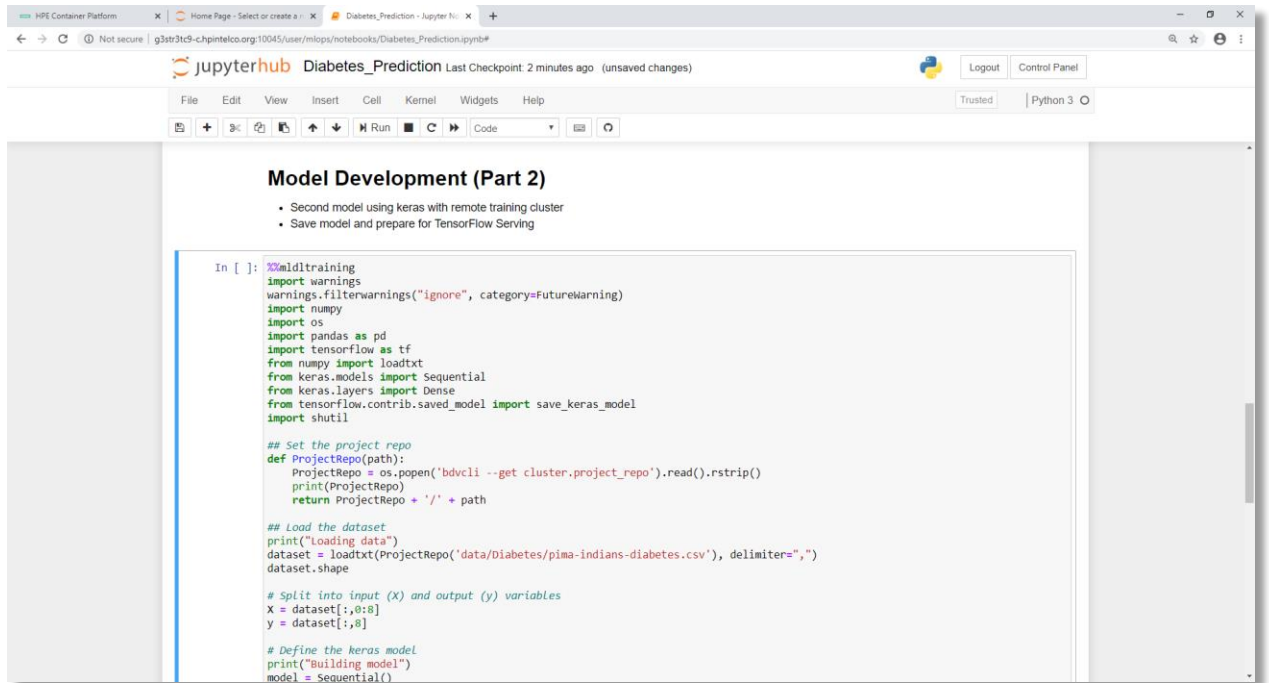
6. Open JupyterHub web ui (Click on NotebookServer)



7. Login in to the Jupyter WebUI as mlops user



8. Open the Diabetes_Prediction.ipynb file, and then run each cell individually.



The screenshot shows a Jupyter Notebook interface with the following content:

Model Development (Part 2)

- Second model using keras with remote training cluster
- Save model and prepare for TensorFlow Serving

```
In [ ]: %%idtraining
import warnings
warnings.filterwarnings("ignore", category=FutureWarning)
import numpy
import os
import pandas as pd
import tensorflow as tf
from numpy import loadtxt
from keras.models import Sequential
from keras.layers import Dense
from tensorflow.contrib.saved_model import save_keras_model
import shutil

## Set the project repo
def ProjectRepo(path):
    ProjectRepo = os.popen('bdvcli --get cluster.project_repo').read().rstrip()
    print(ProjectRepo)
    return ProjectRepo + '/' + path

## Load the dataset
print("Loading data")
dataset = loadtxt(ProjectRepo('data/diabetes/pima-indians-diabetes.csv'), delimiter=",")
dataset.shape

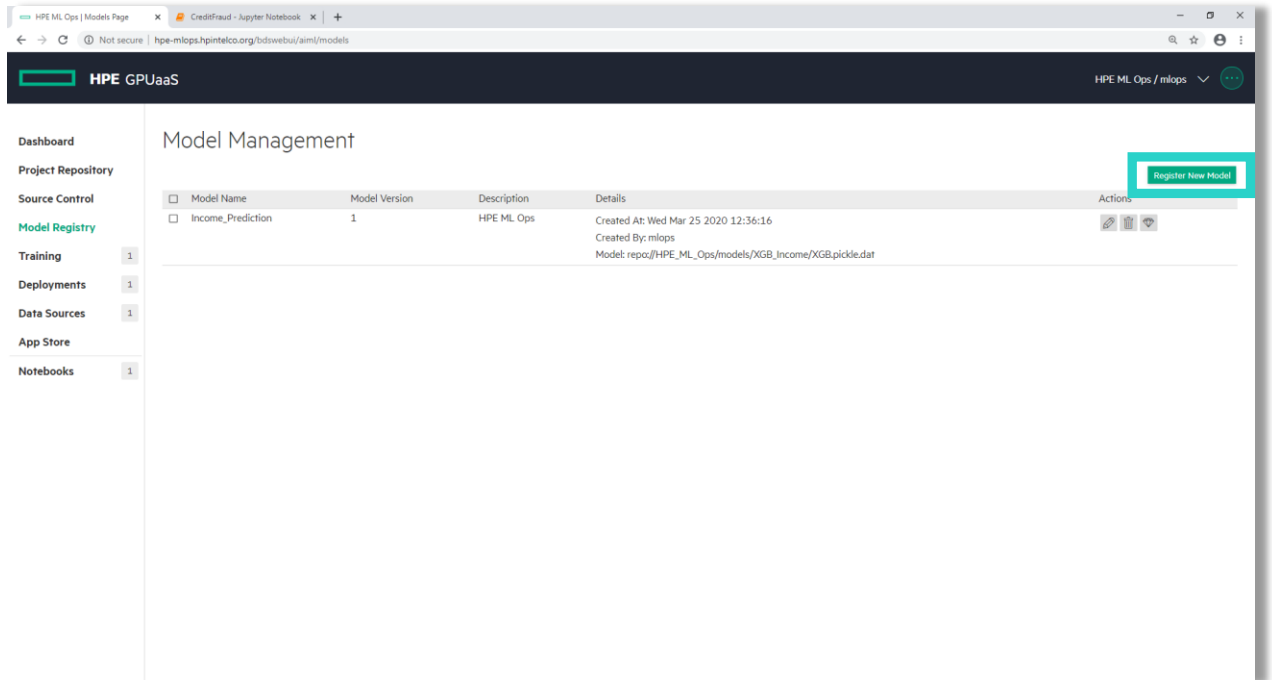
# Split into input (x) and output (y) variables
x = dataset[:,0:8]
y = dataset[:,8]

# Define the keras model
print("Building model")
model = Sequential()
```

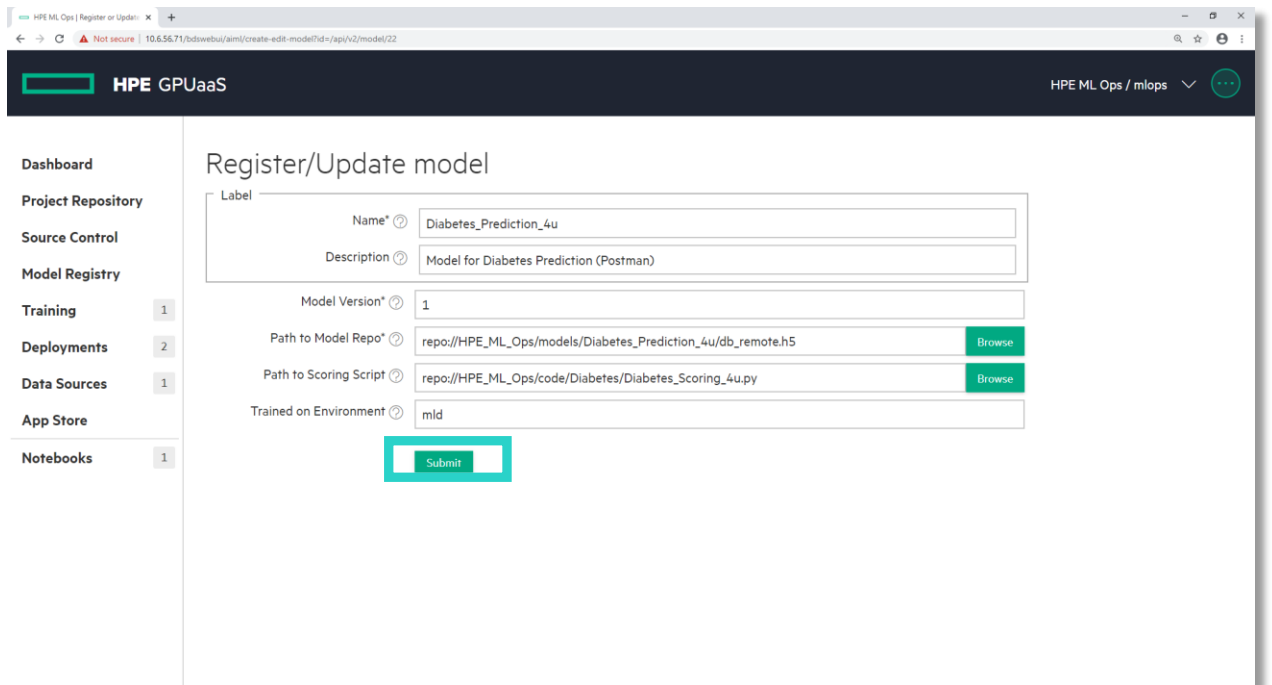
(Caution: Update the variable according to your Training Cluster including History urls.)

Register and Deploy the Model

1. Select Model Registry from the left menu and click on Register New Model






2. Register New Model using the information displayed below.

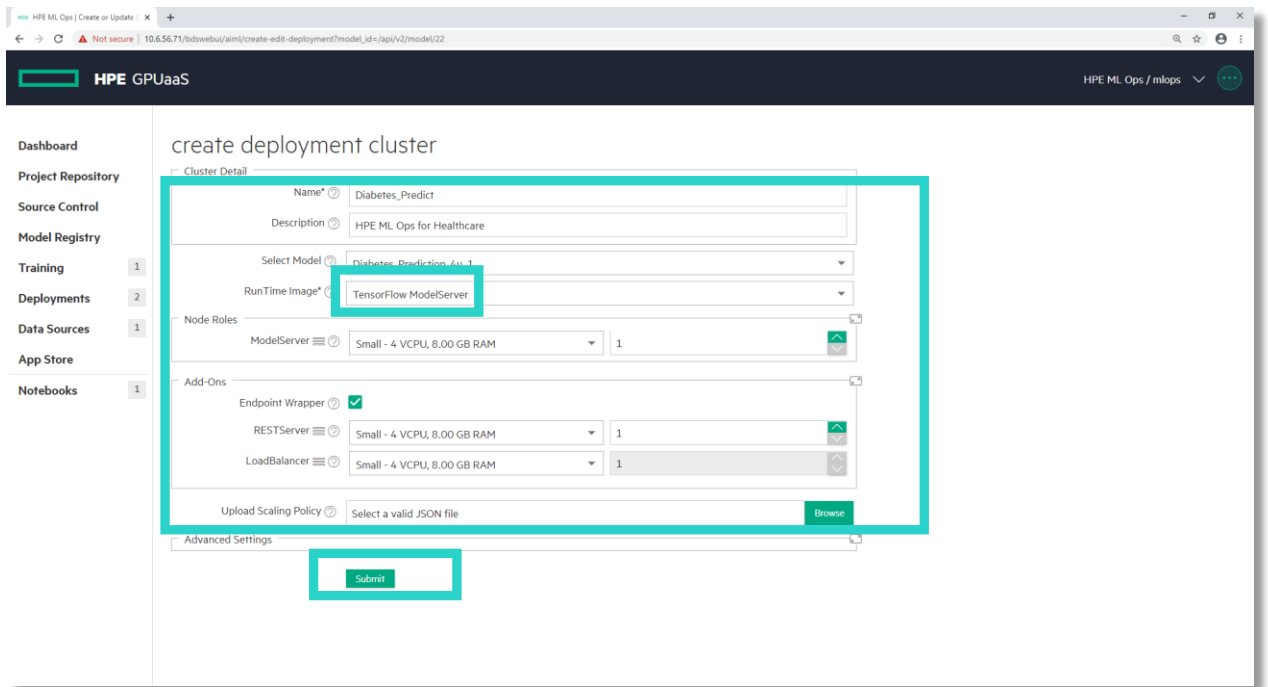


3. Deploy your registered model (Click on Deploy)

Model Management

| | | | | | Register New Model |
|--------------------------|------------------------|---------------|---|--|---|
| <input type="checkbox"/> | Model Name | Model Version | Description | Details | Actions |
| <input type="checkbox"/> | Diabetes_Prediction_4u | 1 | Model for Diabetes Prediction (Postman) | Created At: Tue May 19 2020 16:37:38 Created By: mllops Model: repo://HPE_ML_Ops/models/Diabetes_Prediction_4u/db_remote.h5 |    |

4. Create a Deployment Cluster for Diabetes Prediction (Click on Submit)



create deployment cluster

Cluster Detail

Name* Diabetes_Predict

Description HPE ML Ops for Healthcare

Select Model Diabetes_Prediction_4u_1

RunTime Image TensorFlow ModelServer

Node Roles

ModelServer Small - 4 VCPU, 8.00 GB RAM 1

Add-Ons

Endpoint Wrapper

RETSerVer Small - 4 VCPU, 8.00 GB RAM 1

LoadBalancer Small - 4 VCPU, 8.00 GB RAM 1
















Upload Scaling Policy Select a valid JSON file

Advanced Settings

Don't forget to update the RunTime Image to TensorFlow ModelServer

5. Diabetes Prediction Cluster is now Ready

Deployments

| | | | | | | Create Deployment |
|--------------------------|-------------------------|--|---|--|--------|---|
| <input type="checkbox"/> | Name | Distribution | Role Configurations | Details | Status | Actions |
| <input type="checkbox"/> | pythonmldl | Python ML/DL Toolkit Dependent Distro: Endpoint Wrapper | InferenceEngine(1/Small) RETSerVer(1/Small) LoadBalancer(1/Small) | Created At: Wed May 06 2020 16:31:20 Created By: mllops Attached Model(s):Income_Prediction_1 | ready |      |
| <input type="checkbox"/> | Diabetes_Predict | TensorFlow ModelServer Dependent Distro: Endpoint Wrapper | ModelServer(1/Small) RETSerVer(1/Small) LoadBalancer(1/Small) | Created At: Tue May 19 2020 16:51:11 Created By: mllops Attached Model(s):Diabetes_Prediction_4u_1 | ready |      |
| <input type="checkbox"/> | Diabetes_Prediction_Web | TensorFlow ModelServer Dependent Distro: Endpoint Wrapper | ModelServer(1/Small) RETSerVer(1/Small) LoadBalancer(1/Small) | Created At: Thu May 07 2020 19:06:03 Created By: mllops Attached Model(s):Diabetes_Prediction_1 | ready |      |

Diabetes_Predict

[AIML/Deployment] HPE ML Ops for Healthcare

ready

Node(s) Info ActionScript(s) ServiceStatus Cluster Histories

Cluster Operations

Public Endpoints Actions

| Name | Distribution | Role | Instance IP | Services |
|----------------------|------------------------|--------------|-------------|---|
| bluedata-763.bdlocal | TensorFlow ModelServer | ModelServer | 172.18.0.36 | Tensorflow Model Server : g3str3tc9-c.hpintelco.org:10039 SSH : g3str3tc9-c.hpintelco.org -p 10037 REST API port for tensorflow serving : g3str3tc9-c.hpintelco.org:10038 |
| bluedata-764.bdlocal | Endpoint Wrapper | RETSerVer | 172.18.0.35 | API Server : http://g3str3tc9-c.hpintelco.org:10040 [Auth Token] SSH : g3str3tc9-c.hpintelco.org -p 10041 |
| bluedata-765.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.34 | Model serving request balancer stats API Server : http://g3str3tc9-c.hpintelco.org:10043 [Auth Token] Model Serving LoadBalancer : http://g3str3tc9-c.hpintelco.org:10044/<<model_name>>/<<model_version>>/predict [Auth Token] |

6. Generate your Prediction Request

Diabetes_Predict

[AIML/Deployment] HPE ML Ops for Healthcare

ready

Node(s) Info ActionScript(s) ServiceStatus Cluster Histories

Cluster Operations

Public Endpoints Actions

| Name | Distribution | Role | Instance IP | Services |
|----------------------|------------------------|--------------|-------------|---|
| bluedata-763.bdlocal | TensorFlow ModelServer | ModelServer | 172.18.0.36 | Tensorflow Model Server : g3str3tc9-c.hpintelco.org:10039 SSH : g3str3tc9-c.hpintelco.org -p 10037 REST API port for tensorflow serving : g3str3tc9-c.hpintelco.org:10038 |
| bluedata-764.bdlocal | Endpoint Wrapper | RETSerVer | 172.18.0.35 | API Server : http://g3str3tc9-c.hpintelco.org:10040 [Auth Token] SSH : g3str3tc9-c.hpintelco.org -p 10041 |
| bluedata-765.bdlocal | Endpoint Wrapper | LoadBalancer | 172.18.0.34 | Model serving request balancer stats API Server : http://g3str3tc9-c.hpintelco.org:10043 [Auth Token] Model Serving LoadBalancer : http://g3str3tc9-c.hpintelco.org:10044/<<model_name>>/<<model_version>>/predict [Auth Token] |

Copy the link and check update the port, model name and version

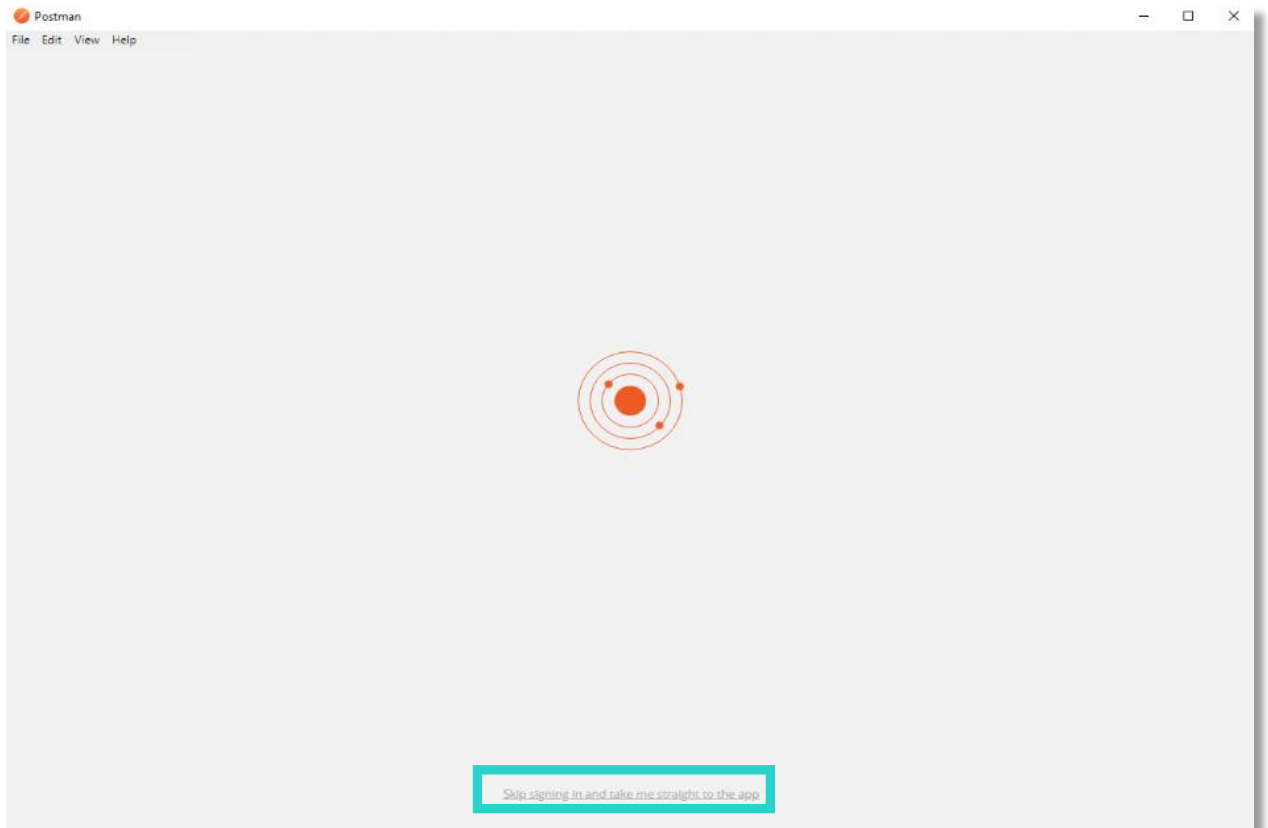
http://g3str3tc9-c.hpintelco.org:10044/Diabetes_Prediction_4u/1/predict
[Auth Token] Click to copy the token

Run the Diabetes prediction using Postman

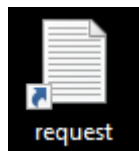


This link will start Postman!

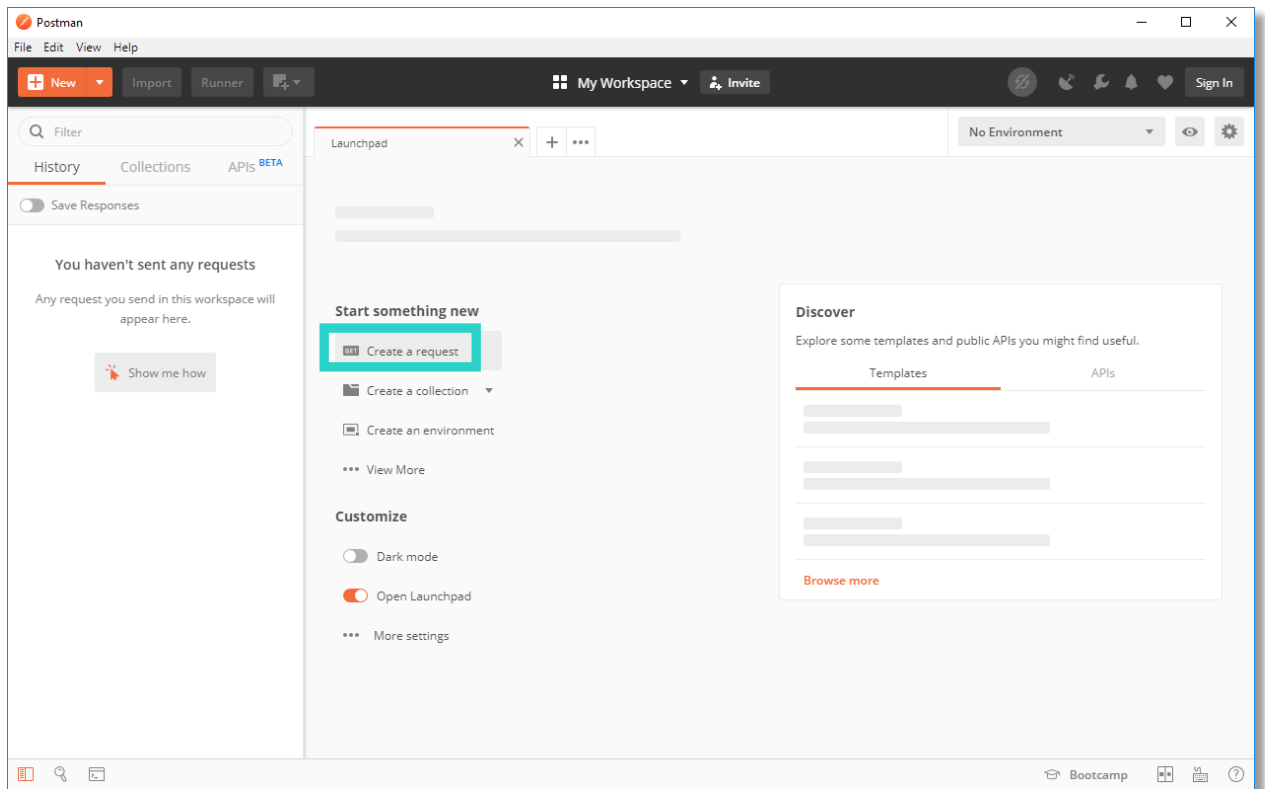
1. Select Skip signing in to go directly to the app



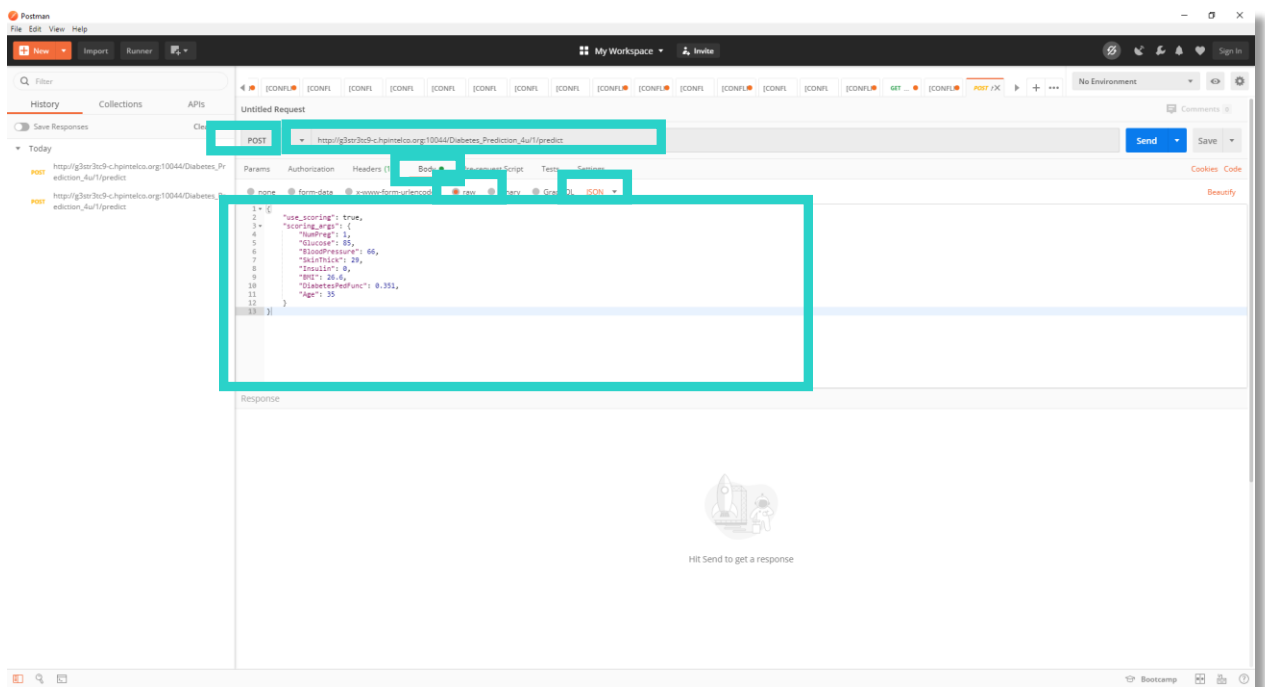
2. Copy the Json request in Postman body tab: Click on below icon



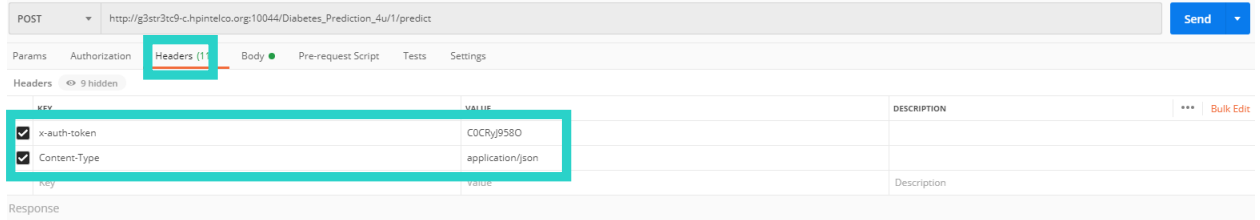
3. Create a new Request



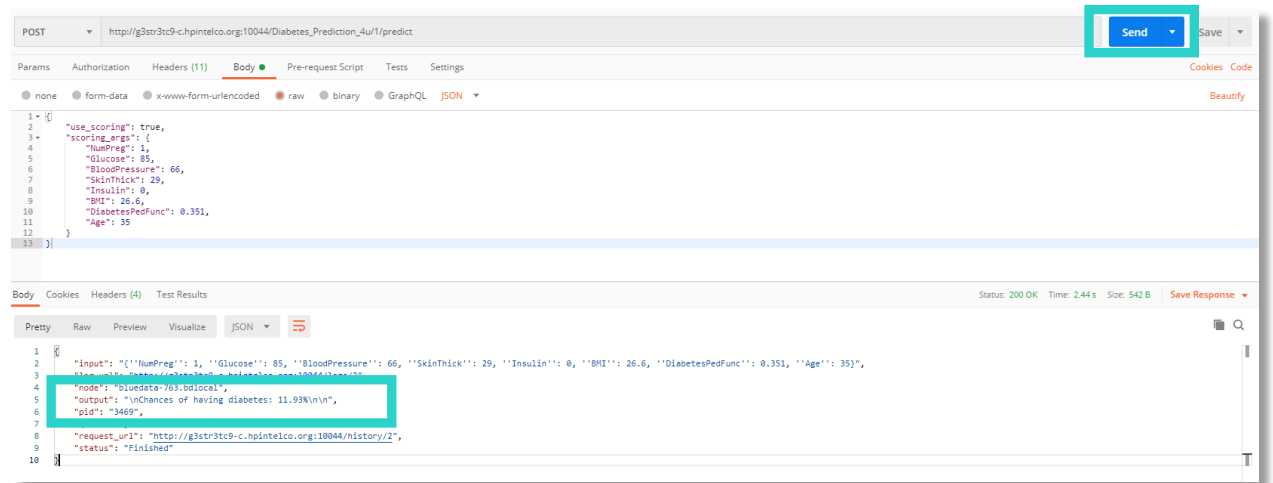
4. Fill in the Body tab to match the image below



5. Fill in the Header tab with your token

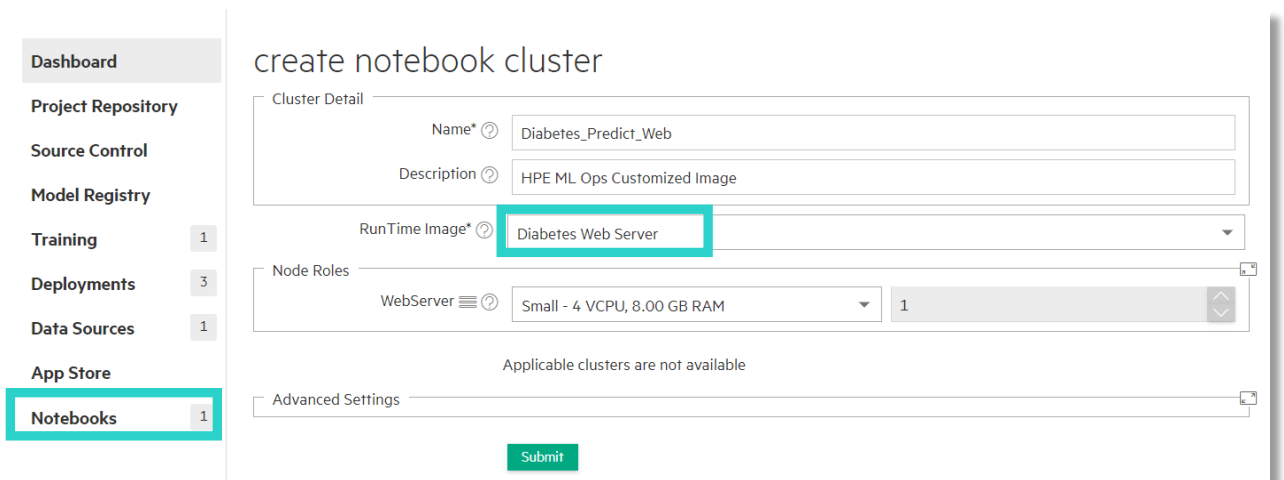


6. Send the post request to the cluster and get the Diabetes prediction result

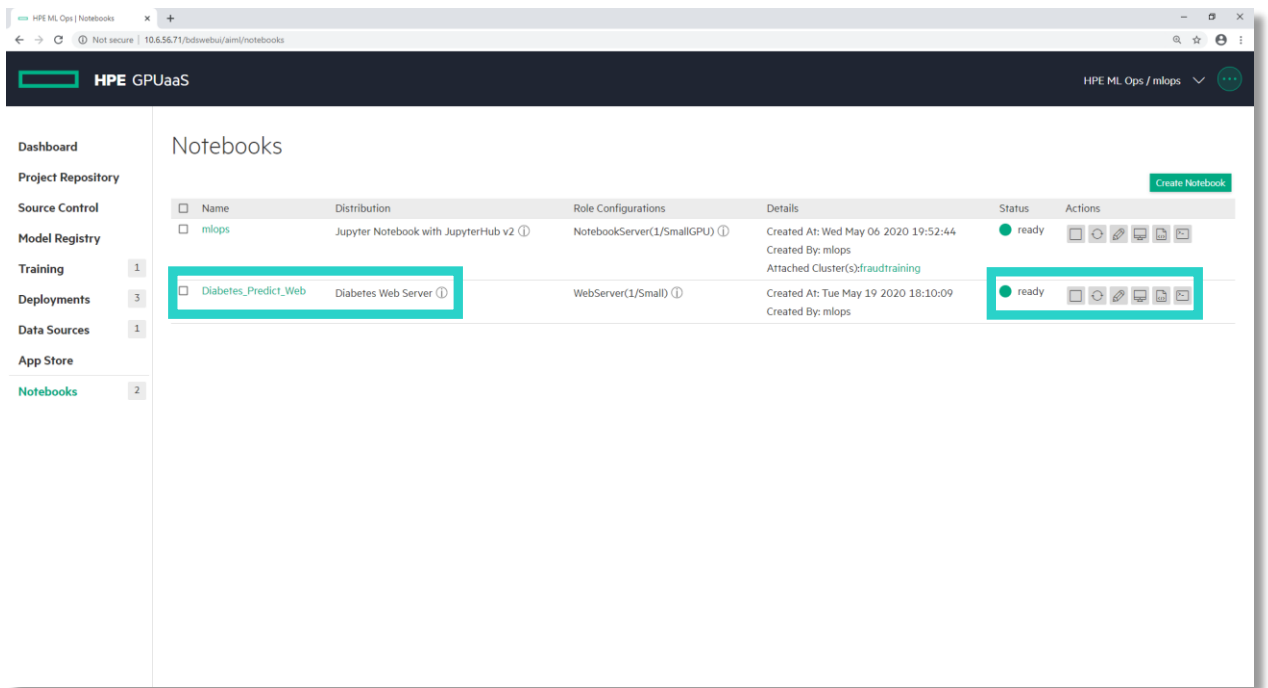


Deploy Customized Notebook Image

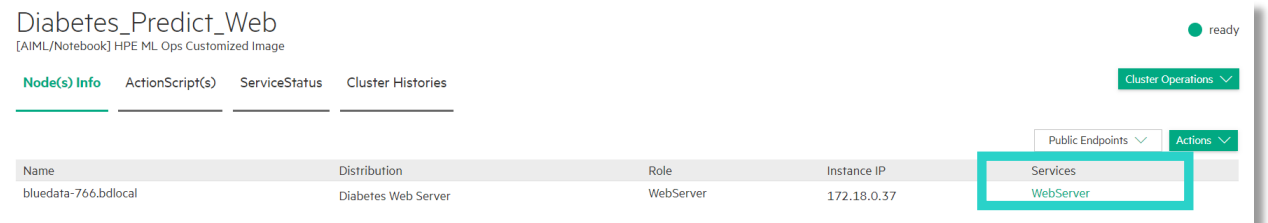
1. Create a new Notebook



2. Diabetes Prediction Notebook is now Ready

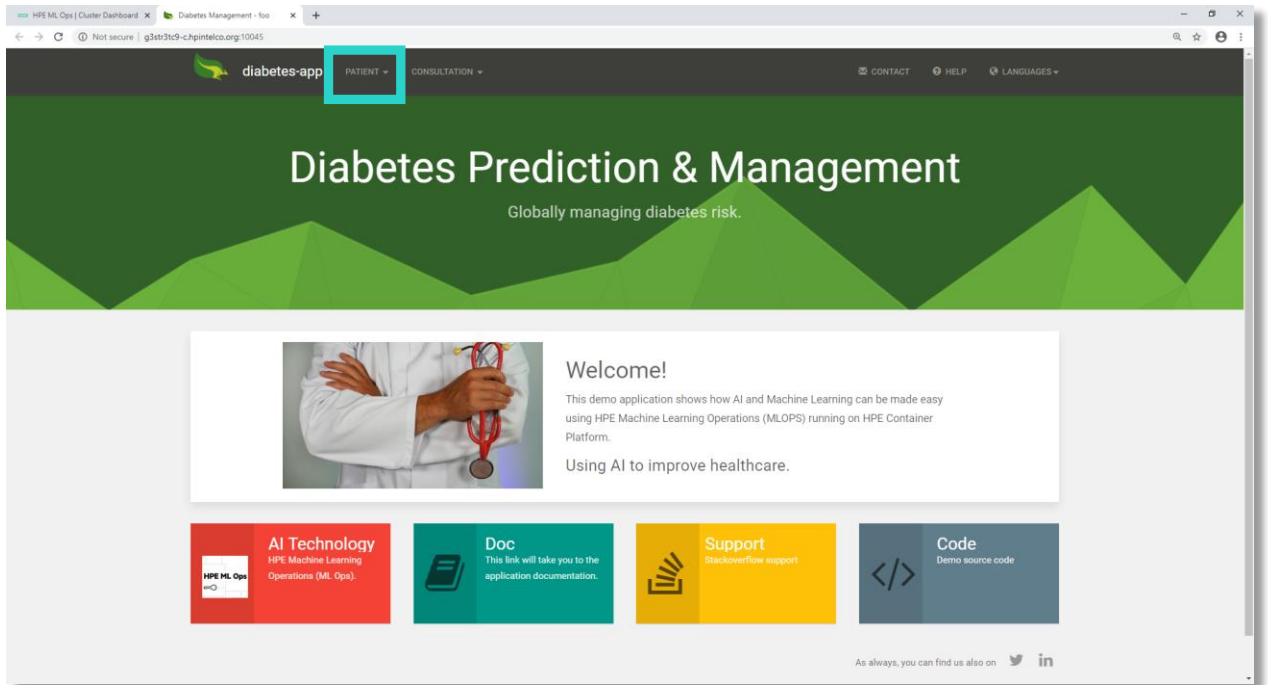


3. Check that the Notebook is ready

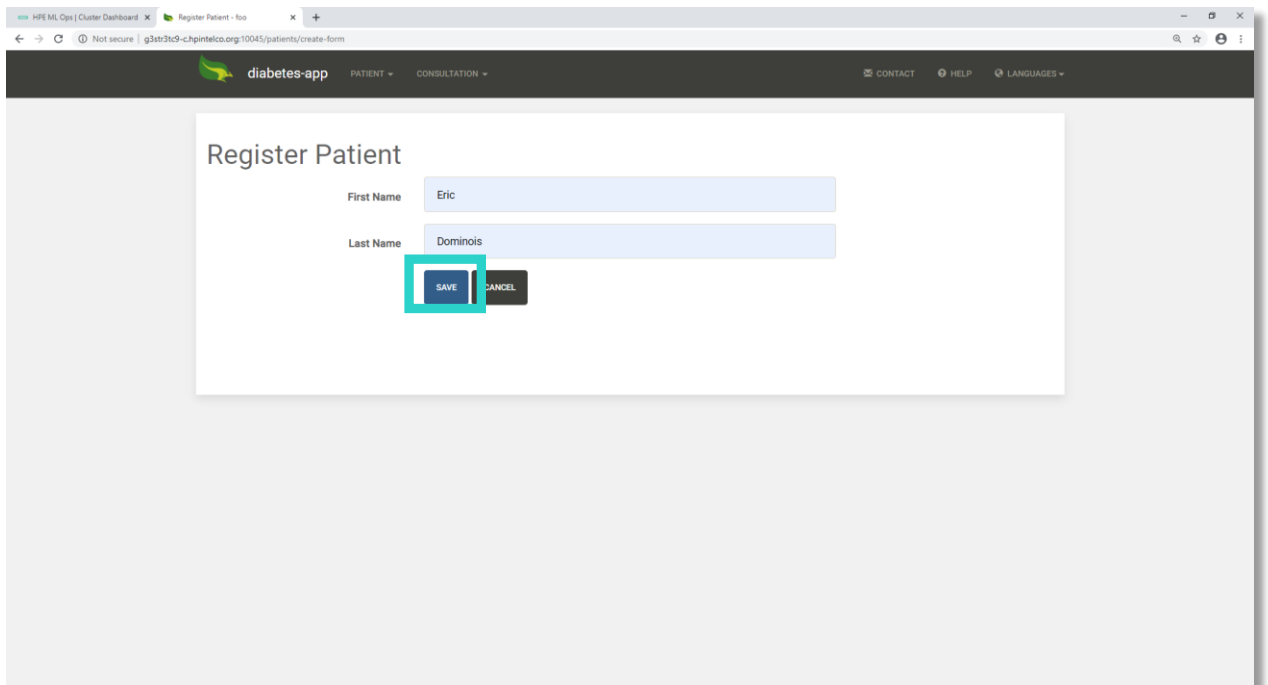


Run the Diabetes prediction using Customized Image

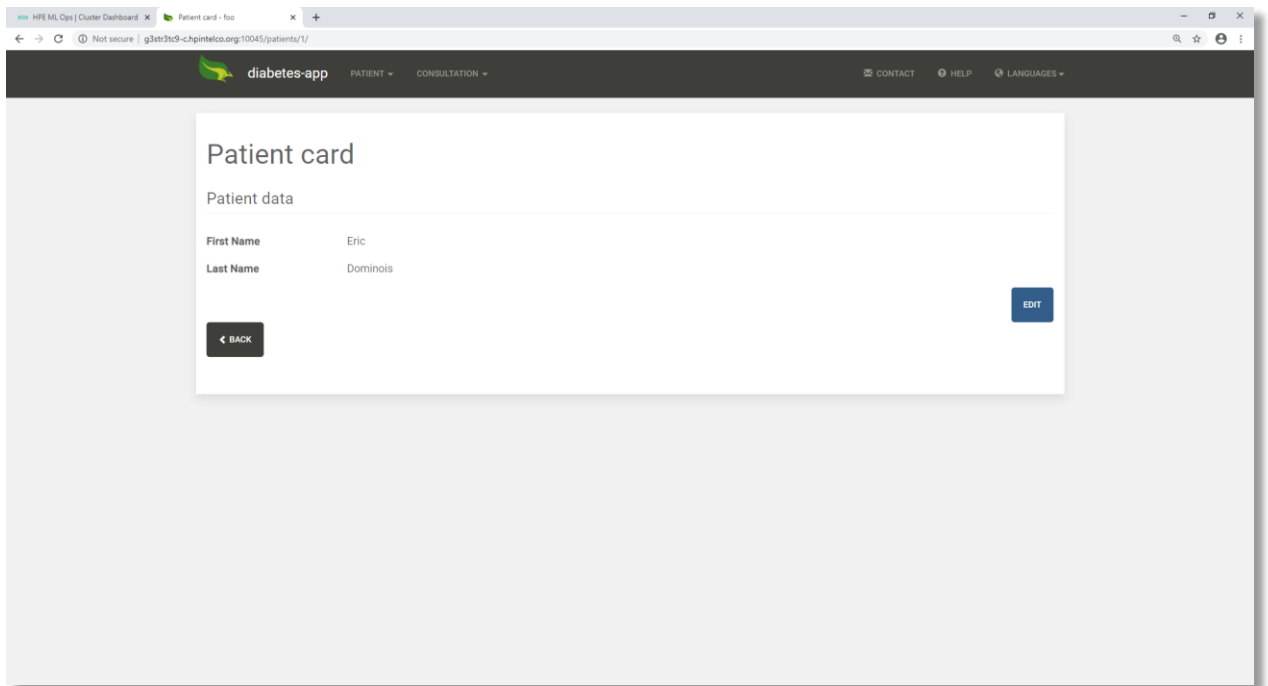
1. Register a new patient



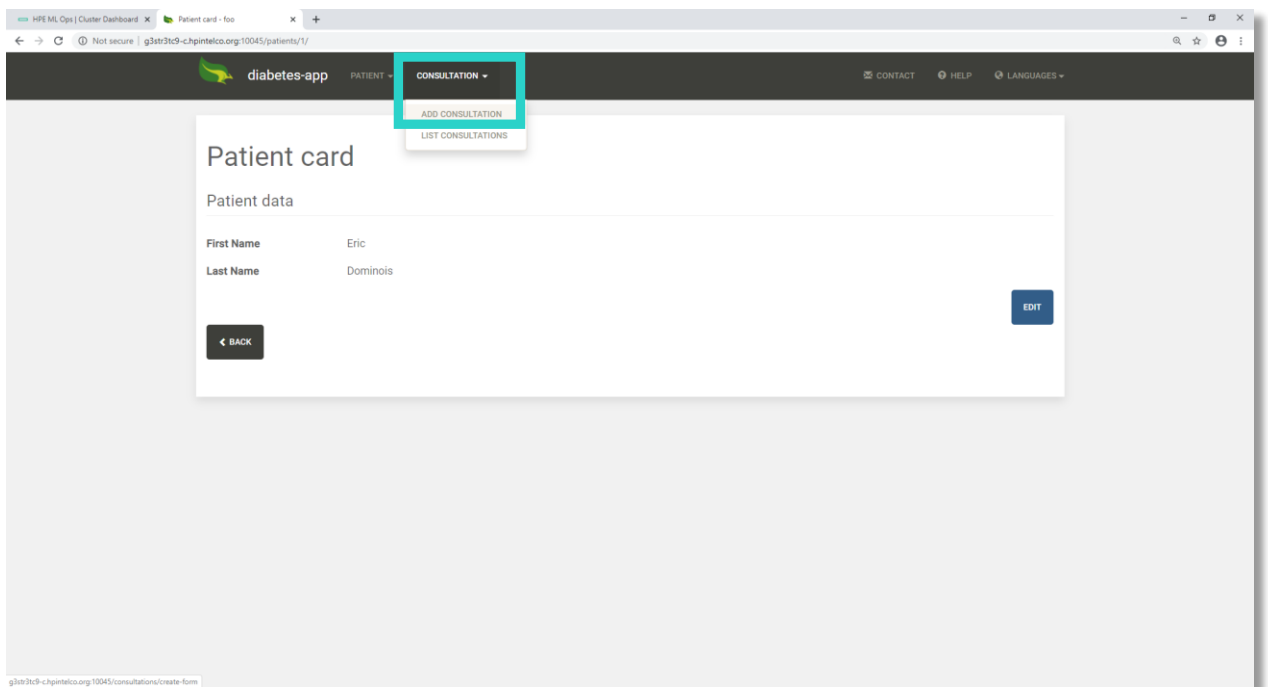
2. Fill in your first and last name and save



3. Patient card has been created



4. Add consultation



5. Create consultation

diabetes-app PATIENT CONSULTATION CONTACT HELP LANGUAGES

Create Consultation

Patient Patient (id='1', version='0', firstName='Eric', lastName='Dominois') x -

Consultation Date May 19, 2020

Pregnancies 1

Glucose 85

Blood Pressure 80

Skin Thickness 29

Insulin 4

Bmi 26.6

Diabetes Pedigree Function 5

Age 60

Prediction % Prediction %

SAVE CANCEL

6. Consultation card has been created with Diabetes prediction score!

diabetes-app PATIENT CONSULTATION CONTACT HELP LANGUAGES

Consultation card

Consultation data

Patient Patient (id='1', version='0', firstName='Eric', lastName='Dominois')

Consultation Date May 19, 2020

Pregnancies 1

Glucose 85

Blood Pressure 80

Skin Thickness 29

Insulin 4

Bmi 26.6

Diabetes Pedigree Function 5

Age 60

Prediction % 26.79

BACK EDIT

Contributors: Terry Chiang, Nanda Vijaydev, and Chris Snow

HPE Container Platform GPU Nodes Monitoring

Environment

Web UI

Username

HPE_ML_Ops

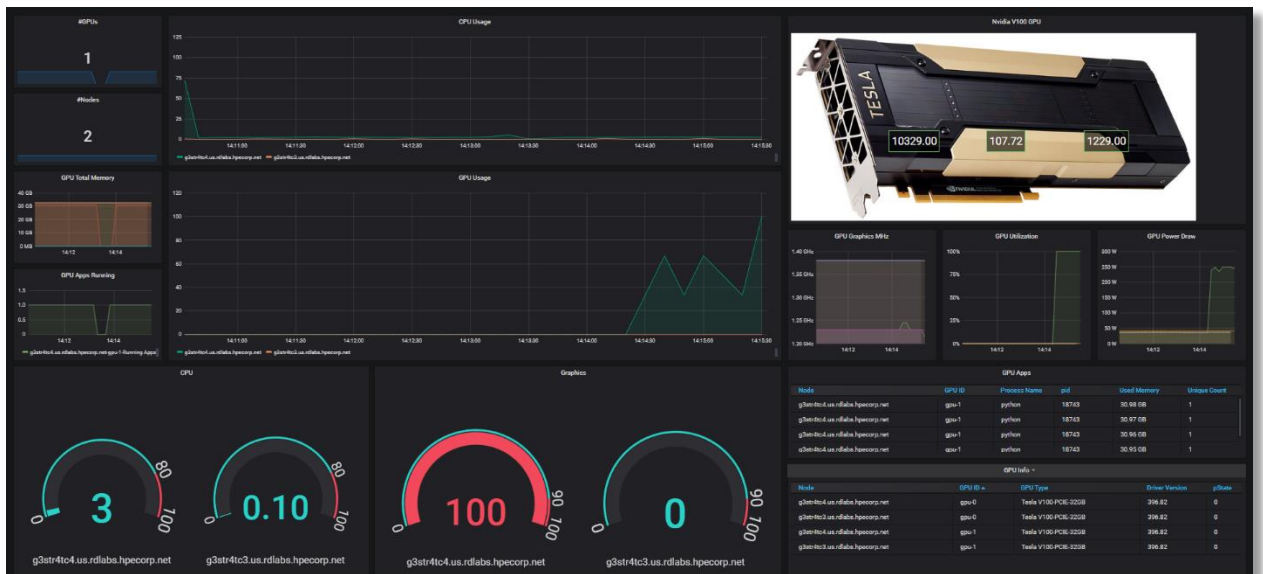
<http://hpe-gpumon.hpintelco.org>

cicdemo@hpintelco.org

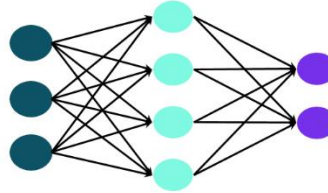
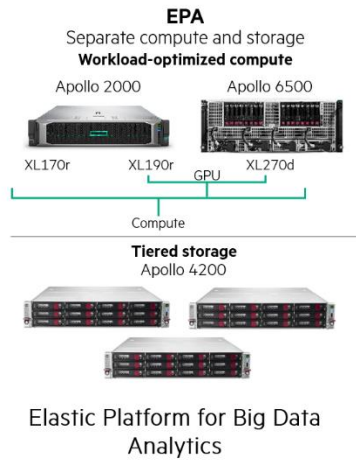
Once connected to the Citrix Jumpost, open in your browser (chrome) the web ui login console, fill the user logon credentials provided on the BlueData_EPIC html page and click "Login".



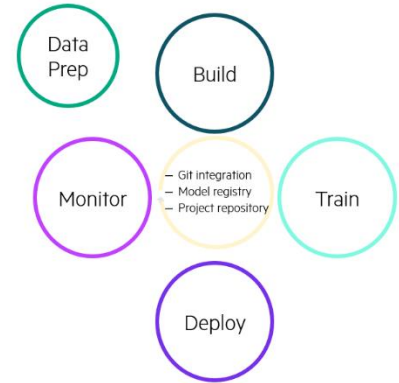
This link will start Chrome and open an html document with links and credentials!



ML/DL ACCELERATE TIME TO VALUE WITH HPE SOLUTIONS



GPU-as-a-Service Solution



Operationalization for the Machine Learning Lifecycle

Thanks!

– End of document –