Hewlett Packard
Enterprise

TAMING STATE/DATA CHALLENGES FOR ML APPLICATIONS AND KUBEFLOW

Skyler Thomas
Distinguished Technologist
skyler.thomas@hpe.com

WHO AM I?
WHAT DO I HOPE YOU WILL LEARN TODAY?

Lead architect for the HPE Ezmeral Data Fabric and HPE Ezmeral ML Ops' Kubeflow support

Although these products can help with many of the challenges I describe today, I will not be talking about these products capabilities. I will have links at the end if you want to check out what we do.

I am hoping we are able to accomplish several things today:

- First, I want to discuss the challenges that occur when you mix data scientists working on machine learning and AI with IT operations
- Next, I will give a brief overview of the Kubeflow workflow and the various state and data challenges that exist in Kubeflow
- Finally, I will dive a bit deeper into specific challenges you will face building these applications and suggest tips for overcoming them.

I will then answer your questions...

At the end of the session, I hope when you leave this session that you will be a more prepared to face the state and data challenges Kubeflow presents and have a few ideas in your back pocket to solve them

DATA SCIENTISTS AND IT HAVE CONFLICTING GOALS!

Limit the hardware you use!

No! you can't see THAT data!

We have standard applications we use here!

Standard nodes!

Virtualize!

Share!

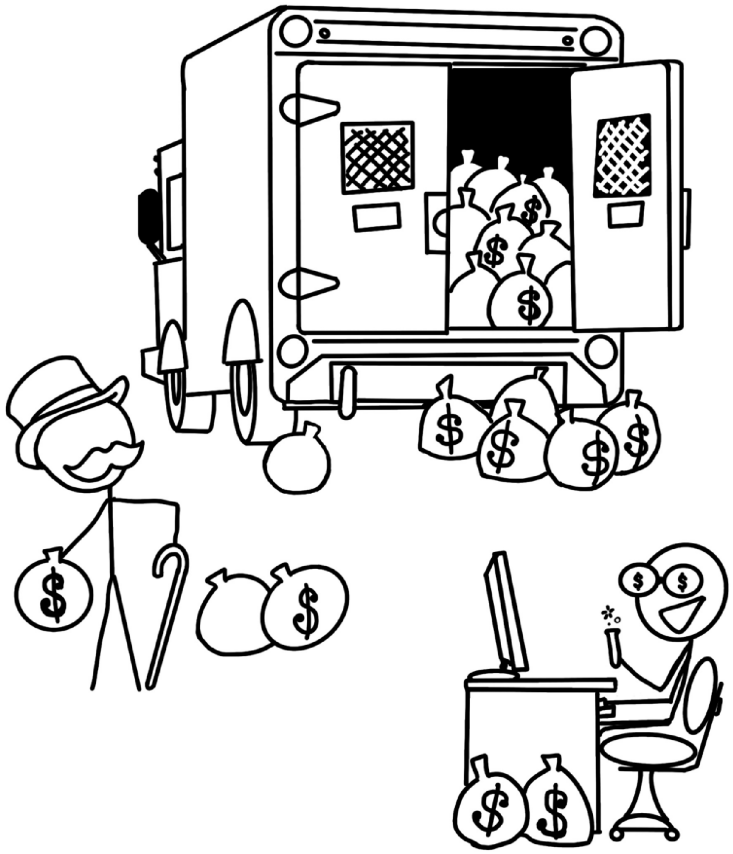
We require a security review before you can run that!

DATA SCIENTISTS VS IT

Antithetical Goals

- Data Scientists want custom hardware - high memory, GPU's and TPU's. IT wants cookie cutter commodity machines wherever possible
- Data Scientists want all the hardware they can get. IT wants to encourage efficiency and sharing
- IT wants to standardize applications and reduce license costs. Data Science wants to use a variety of applications and frameworks
- IT wants to ensure software is safe and supported. Data Science wants cutting edge tools from academia
- IT wants to carefully categorize and restrict data for security and regulatory reasons. Data science wants to have access to as much raw data as possible for training their models.

ISOLATION IS THE SOLUTION THAT HAS WORKED BEST UNTIL NOW



* Kubernetes doesn’t mean isolation is dead. Ephemeral data scientist Kubernetes compute environments can still make sense for various reasons. Just be careful about data gravity challenges..

TRADITIONAL SOLUTION - ISOLATION

There has been no good way to solve these issues except for isolation

The most successful method has been to back a truck full of money up to data science to create their own environments with tons of memory and GPU's

Hand rolled every time

Cloud has helped but TPU and GPU environments have unique challenges

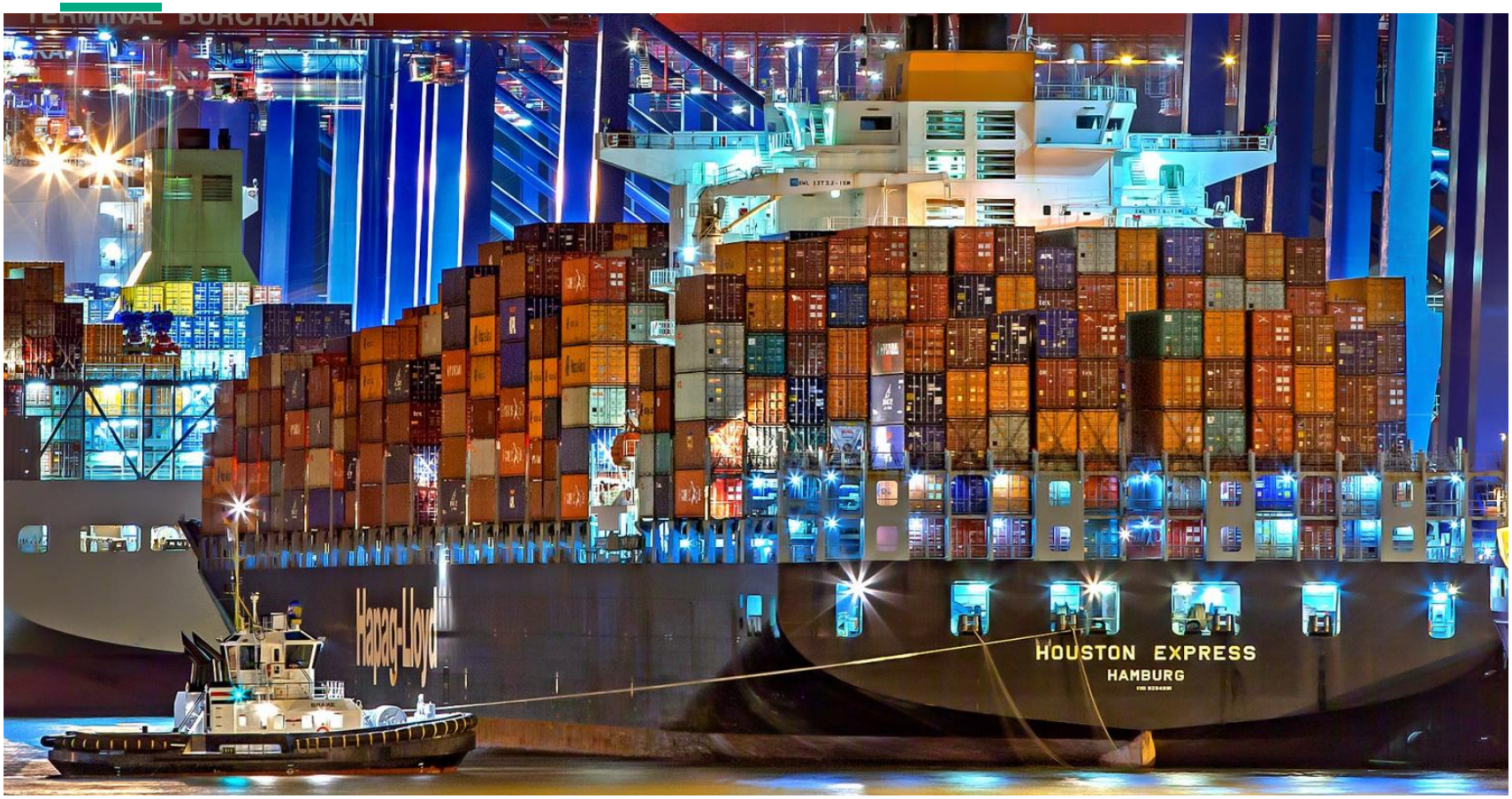
BIGGEST LESSON FROM BIG DATA, SPARK, AND TENSORFLOW

BIGGEST LESSON FROM BIG DATA, SPARK, AND TENSORFLOW ON BARE METAL?

Manual environment creation is expensive and time consuming

Standing up new environments for each set of tasks does not work

ENTER KUBERNETES



KUBERNETES

Kubernetes solved several critical problems with traditional ML or Big Data environments

- Hardware does not need to be dedicated to data science
- Complex scheduling allows different node types to be used (GPU/TPU high mem vs normal nodes)
- App containerization means many more types of apps can be support

However, Kubernetes has a number of challenges fitting into the data science world

The name Data science is a clue

Machine learning requires ton of data or state to work

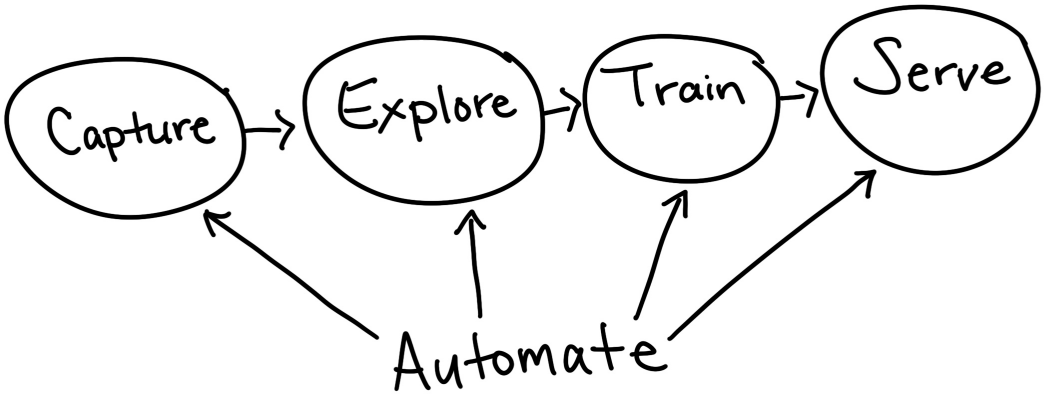
Kubernetes was initially designed for stateless apps.

Over time handling state in applications have improved on Kubernetes… We have things like:

- Persistent Volumes and Storage Classes
- CSI
- Volume based scheduling
- And others

But, there are still MAJOR challenges

KUBEFLOW WORKFLOW



There are different types of state that Machine Learning applications need to deal with. Each type of data poses its own set of challenges

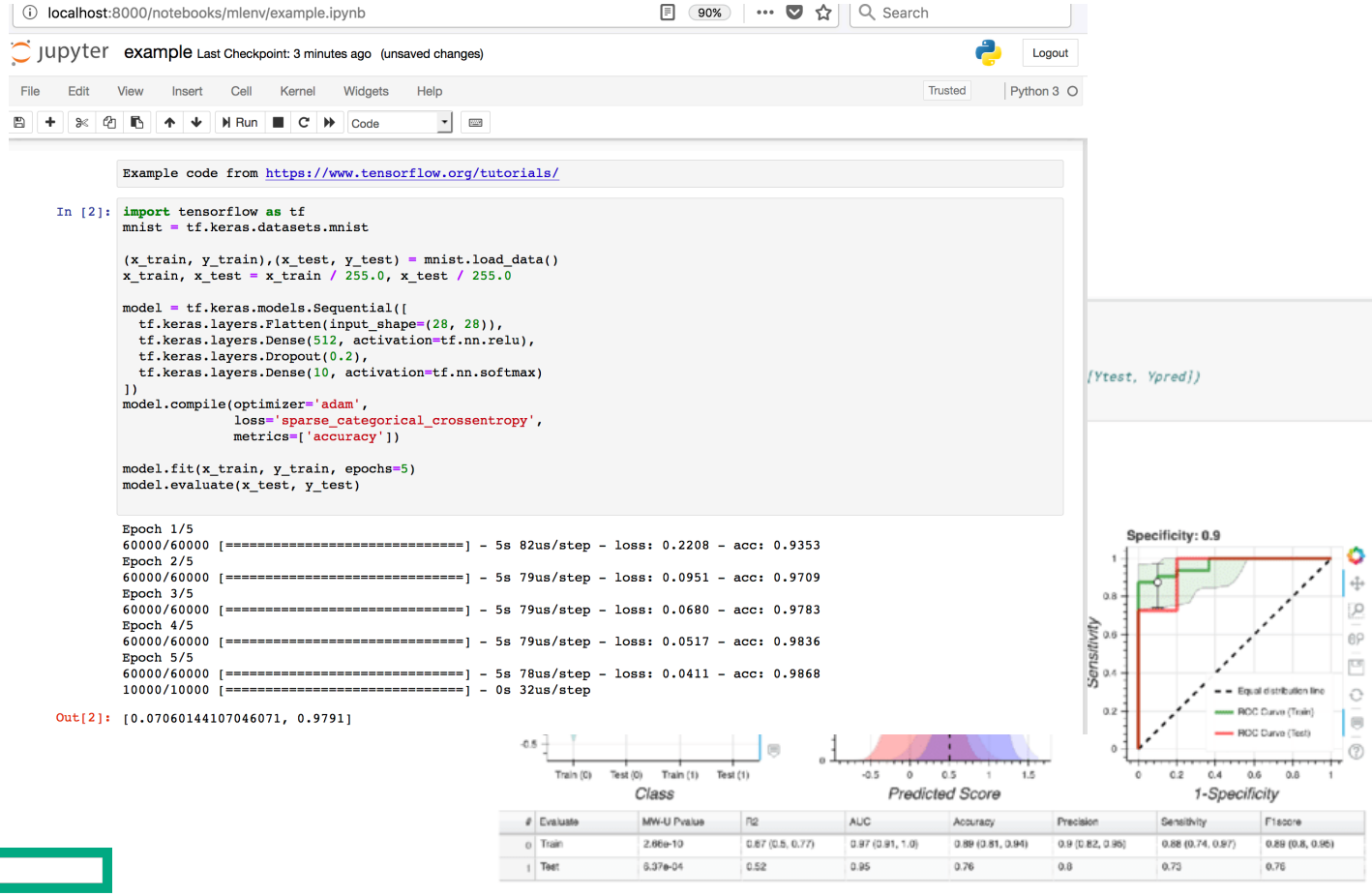
STATE





Notebooks	Training Data	Libraries	Models	Logs
How can we share notebooks between team members?	We can might need to handle hundreds of Petabytes of Training Data. How do we move this close to compute? How do we protect privacy if some of this data is sensitive? How do we follow legal requirements?	How do we avoid teaching every data scientist to become a containerization expert just so they can use a particular python library in their training code?	How do we register, query and create model files? How do we auto-promote models for test and serving?	Many Distributed Training solutions generate logs that are important to read by data scientists?

NOTEBOOKS

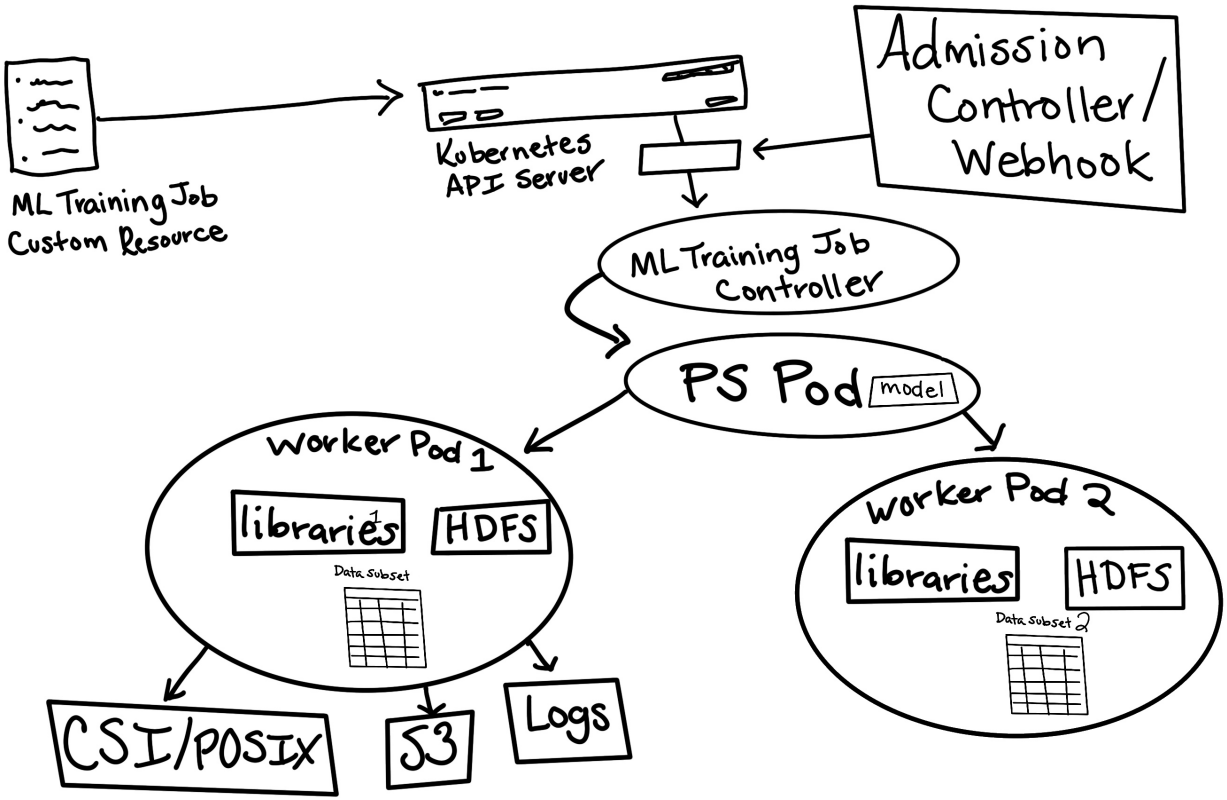

localhost:8000/notebooks/mlenv/example.ipynb
jupyter example Last Checkpoint: 3 minutes ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3
Run
Code
Example code from https://www.tensorflow.org/tutorials/
In [2]:
import tensorflow as tf
mnist = tf.keras.datasets.mnist
(x_train, y_train),(x_test, y_test) = mnist.load_data()
x_train, x_test = x_train / 255.0, x_test / 255.0
model = tf.keras.models.Sequential([
tf.keras.layers.Flatten(input_shape=(28, 28)),
tf.keras.layers.Dense(512, activation=tf.nn.relu),
tf.keras.layers.Dropout(0.2),
tf.keras.layers.Dense(10, activation=tf.nn.softmax)
])
model.compile(optimizer='adam',
loss='sparse_categorical_crossentropy',
metrics=['accuracy'])
model.fit(x_train, y_train, epochs=5)
model.evaluate(x_test, y_test)
Epoch 1/5
60000/60000 [==============================] - 5s 82us/step - loss: 0.2208 - acc: 0.9353
Epoch 2/5
60000/60000 [==============================] - 5s 79us/step - loss: 0.0951 - acc: 0.9709
Epoch 3/5
60000/60000 [==============================] - 5s 79us/step - loss: 0.0680 - acc: 0.9783
Epoch 4/5
60000/60000 [==============================] - 5s 79us/step - loss: 0.0517 - acc: 0.9836
Epoch 5/5
60000/60000 [==============================] - 5s 78us/step - loss: 0.0411 - acc: 0.9868
10000/10000 [==============================] - 0s 32us/step
Out[2]: [0.07060144107046071, 0.9791]
[Ytest, Ypred])
Specificity: 0.9
Sensitivity
1-Specificity
Equal distribution line
ROC Curve (Train)
ROC Curve (Test)
Class
Predicted Score
Train (0)
Test (0)
Train (1)
Test (1)
Evaluate MW-U Pvalue Π2 AUC Accuracy Precision Sensitivity F1score
0 Train 2.66e-10 0.67 (0.5, 0.77) 0.97 (0.91, 1.0) 0.89 (0.81, 0.94) 0.9 (0.82, 0.95) 0.88 (0.74, 0.97) 0.89 (0.8, 0.95)
1 Test 9.37e-04 0.52 0.95 0.76 0.8 0.73 0.76

NOTEBOOKS

Notebooks are where data scientists live most of the time

- Container based
- Usually Jupyter or Zepplin (if Spark app)
- Normally run as a service account
- Examine and scrub data
- Experiment and visualize data
- Create toy models
- Launch large training jobs and automations

DISTRIBUTED TRAINING

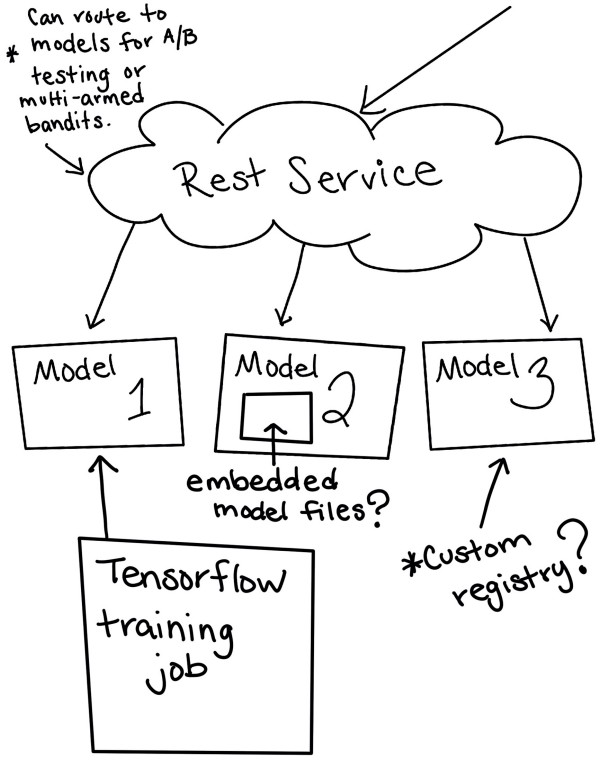


DISTRIBUTED TRAINING

Most Kubeflow and other ML Training technologies follow a similar “Job Operator” pattern. This includes: Spark, TensorFlow, and PyTorch

- A ML Job Custom Resource of some type is submitted via Kubectl or other client tool to the Kubernetes API Server
- A ML Job Controller of some type sees the submission and then dynamically creates a set of control or parameter server pods and a set of worker pods
- The worker pods retrieve/manipulate training data via api’s like CSI/POSIX, S3/Object Storage, or HDFS
- A model and logs are generated
- The job completes and the pods are torn down
- Tip: Anytime a custom resource type is submitted to the Kubernetes API server is an opportunity to use an admission controller webhook to intercept the request and create various resources in the appropriate namespace. This can help significantly with ensuring the proper secrets, ConfigMaps, or PersistentVolumes are created

MODEL SERVING



MODEL SERVING

Model serving typically doesn't require training data but there are still some challenges from a state perspective

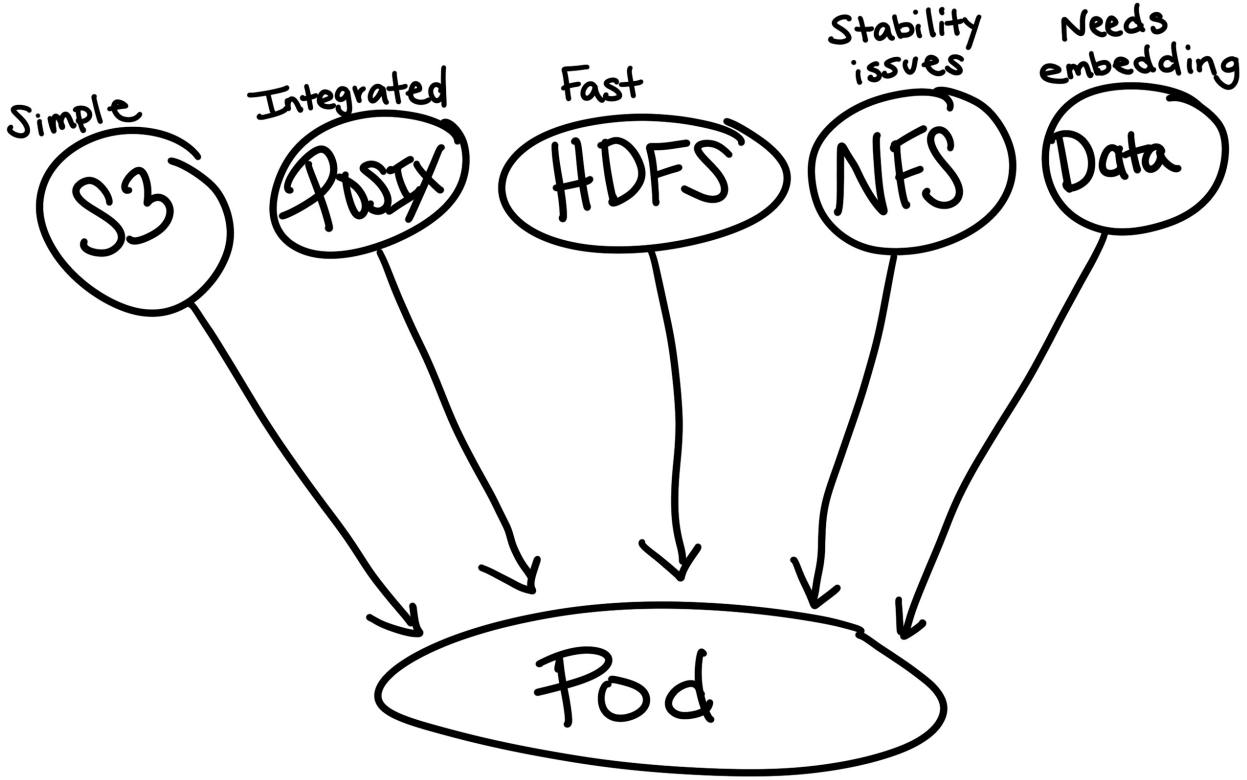
Models created during distributed training must also be available for serving

Are the model files inserted into a new container or is it mounted as a volume?

Is the model container moved into production via copying using a workflow engine like Argo?

Where are the containers stored? A shared docker registry or a private registry? How go you ensure that only approved models in the approved registry are served by model serving infrastructure? (Seldon, Tensorflow Serving, KFServing)

CHALLENGE: DATA API AVAILABILITY



CHALLENGE: DATA API AVAILABILITY

Unlike Hadoop, ML tools are not unified around a single data access API. Different ML applications require different storage API's

OBJECT STORAGE/S3 – USEFUL API FOR SIMPLICITY/CAN FIT INTO A PIPELINE UI VERY EASILY. CAN HAVE ISSUES WITH SPEED & THROUGHPUT. LONG TERM WORK INTEGRATING INTO KUBERNETES

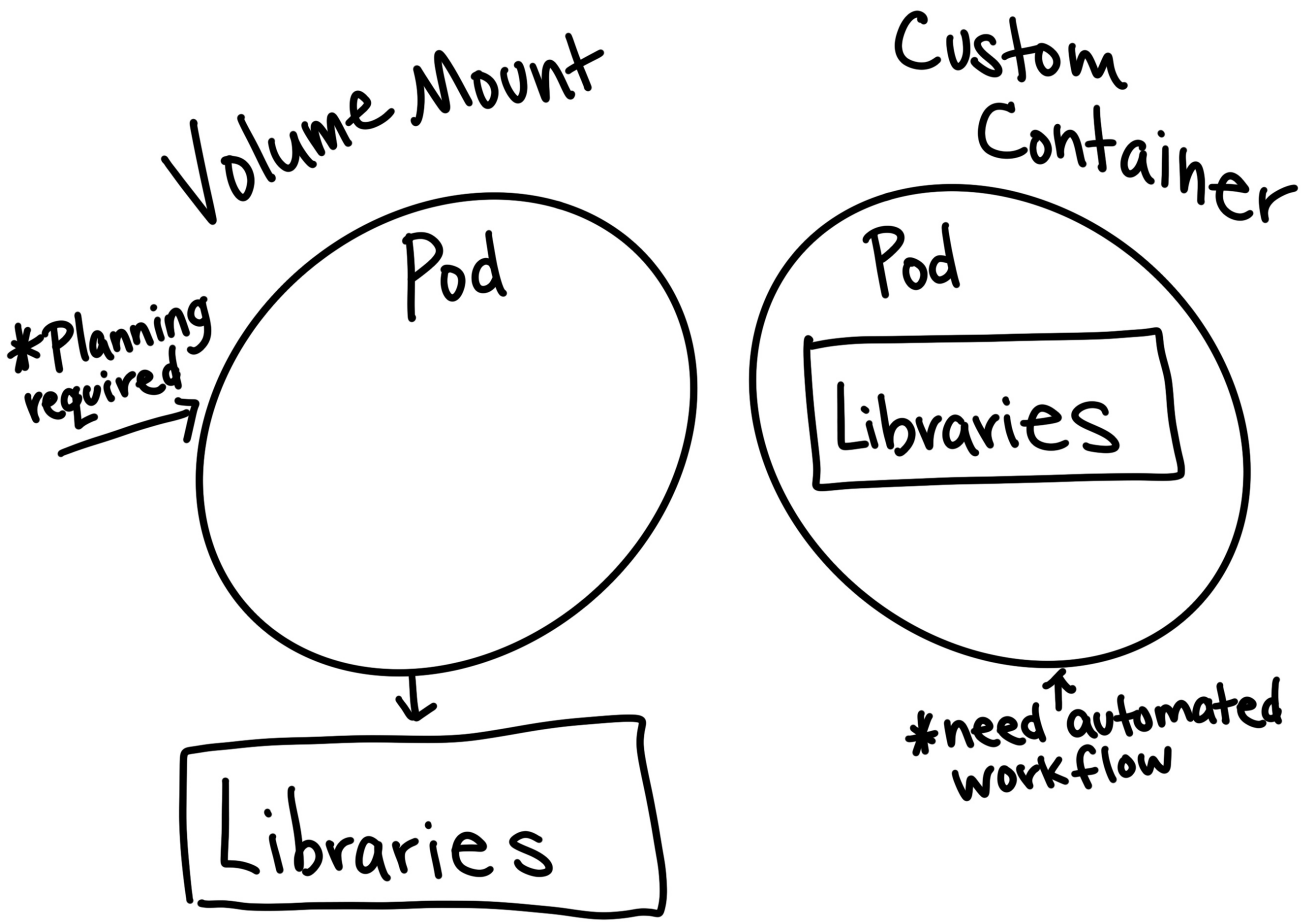
POSIX – USEFUL FOR SPEED. FITS KUBERNETES MODEL VIA CSI NICELY. UNLESS CSI DRIVER IS FOR DISTRIBUTED FILE SYSTEM, COPYING DATA IS CHALLENGING

HDFS – USEFUL FOR RAW THROUGHPUT. LIMITED SUPPORT FROM NON-HADOOP HERITAGE TOOLS. NOT PLATFORM INTEGRATED/MUST BE EMBEDDED IN CONTAINERS

NFS – SLOW. DIFFICULT TO SUPPORT EXTREME THROUGHPUT. STABILITY ISSUES. REQUIRES CONTAINER EMBEDDING

VARIOUS DATABASE PROTOCOLS – REQUIRES EMBEDDING IN CONTAINERS

CHALLENGE: LIBRARY STATE



CHALLENGE: LIBRARY STATE

Each training job or notebook may need different libraries or other assets

Two major schools of thought

- Build container for each task
- Mount a volume for libraries or assets into a common container

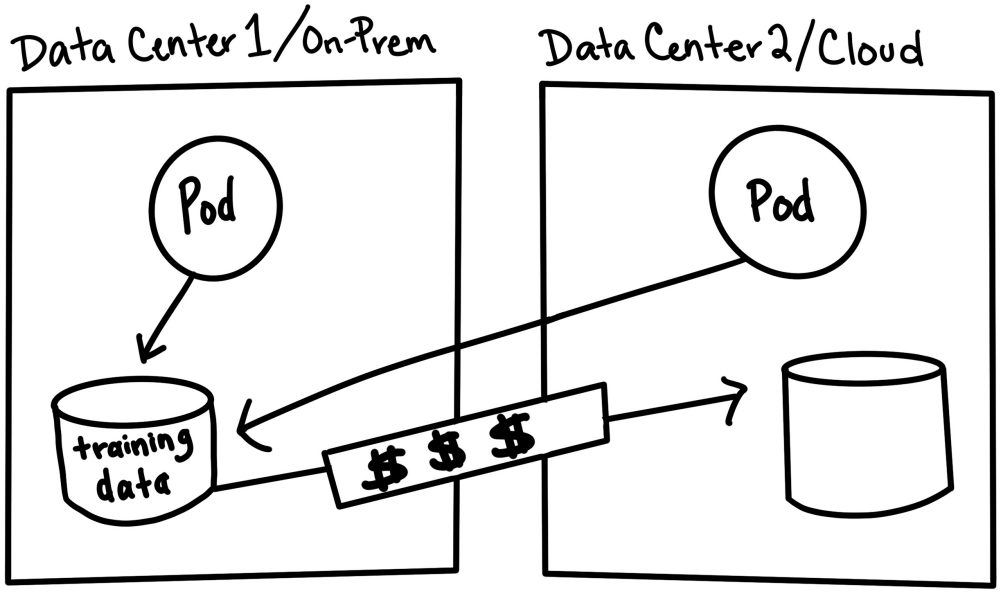
The container route has several downsides

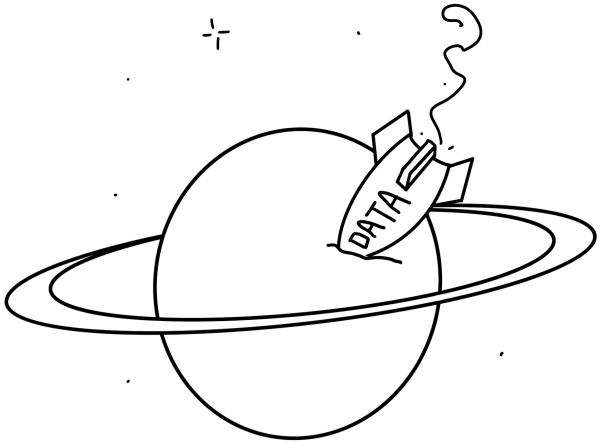
- Data scientist must be more skilled to some degree with container creation
- Roll your own container security issues
- Container management and storage

Volume mounting requires more time and planning. Probably requires a CSI provider with a distributed file system

Automate container creation through pipelines or a workflow engine like Argo. Recipes or container scripts break down quickly

CHALLENGE: TRAINING DATA GRAVITY





CHALLENGE: TRAINING DATA GRAVITY

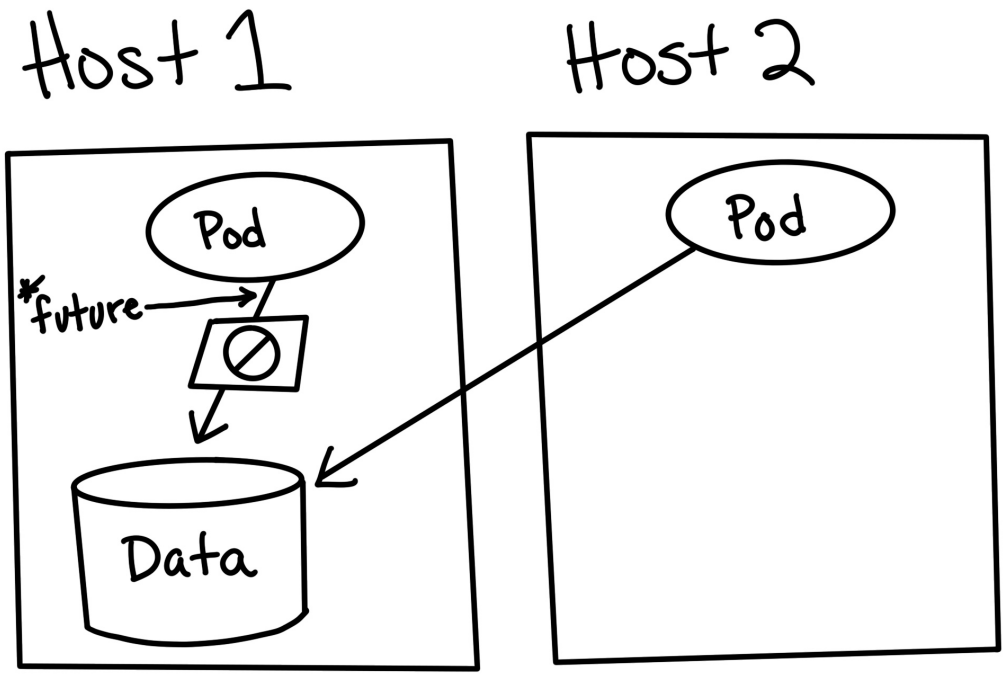
Data Gravity is the idea that its hard to move data around and that data naturally attracts compute closer to itself. There are a number of reasons for data gravity:

- Time and cost to move data. We have customers building self driving simulation environments. They have hundreds of Petabytes of sensor data that training pods must access. It is very difficult to copy this data somewhere
- Government Regulations. Many countries require data to remain inside their borders
- Privacy Concerns. Many companies or governments restrict where sensitive data can be moved or how that data can be accessed.

Some data gravity tips:

- When data must live outside the cloud and within a corporate firewall, you must choose between allowing external compute to access the data or creating an on-prem Kubernetes environment. In a large number of cases, the on-prem solution is superior for performance, security, or legal reasons
- In many cases, spinning up quick Kubernetes solutions in the cloud may be useful to handle the worries about Data Scientists rolling their own containers. Consider a shared cloud-based data lake for the ephemeral data scientist Kubernetes environments

CHALLENGE: TRAINING DATA LOCALITY



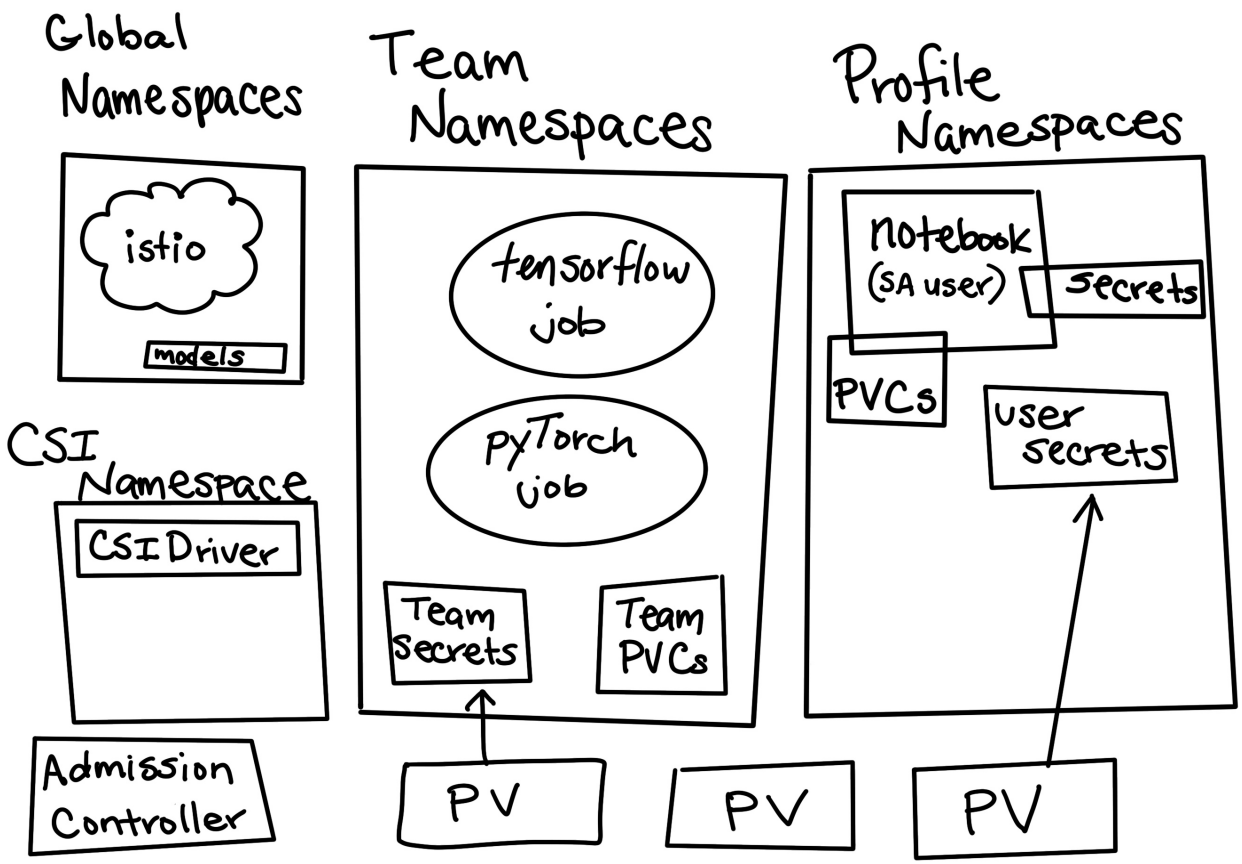
CHALLENGE: TRAINING DATA LOCALITY

The Hadoop world spent an extreme amount of time ensuring data locality and optimizing the HDFS protocol for this type of access.

As network speeds have improved, true data locality has become less important. In many cases fast POSIX access to a quality distributed file system is enough

Block local storage, scheduler improvements, and custom schedulers may all make true data locality something that is reasonable for Kubernetes ML frameworks in the future. For instance, the Spark project is looking closely at how to do this with HDFS. However, at this point, true data locality for machine learning is more of a future goal rather than reality.

CHALLENGE: DATA SECURITY



CHALLENGE: DATA SECURITY

- Data security is one of the most complex subjects for machine learning applications and Kubernetes.
- We will not be able to cover all the nuances in this presentation but I want to give you a feel for where the issues come from and a few tips. At some point, much of the pain will have to be shouldered by Kubernetes ML environment Vendors because it is really challenging to do things right.
- There are two main causes for the difficulties:
  - Notebook Service Accounts
  - Namespaces
- If you think about data, there are three types of data:
  - Globally Shared Data – This is data shared between multiple teams, users, and projects
  - Team Shared Data – This is data shared between team members
  - User Data – This is data for a single data scientist

CHALLENGE: DATA SECURITY & VOLUMES

- CSI, PVs, and StorageClasses
  - CSI uses secrets with storage backend credentials. These are attached to PV's or StorageClasses to pass to the driver so it can access data securely.
  - Individual Secrets are difficult to protect in a namespace shared by multiple users.
  - This means its tricky to use PV's for user data in a shared namespace.
  - StorageClasses are global so its tricky to use user specific
- Tip: Plan your namespaces and data owners VERY carefully. Think of the security implications of where your secrets have to be located

CHALLENGE: DATA SECURITY & TEAM RESOURCES

- Kubeflow has the notion of “profile” namespaces for individual users. This is where user notebooks run. Unfortunately, this may make it challenging for your users to access team data in a notebook.
- Most vendors have the notion of Tenant, Team, or Project namespaces
- Most organizations want to manage their Kubernetes resource consumption via quotas at a team level. This means team/tenant namespaces should be created. Distributed training jobs should run in those namespaces
- Tip: Use webhooks (admission controllers), Istio, and JupyterHub to create resources like volumes and storage classes in “profile” namespaces for team volumes.
- Tip: Training jobs that run in team namespaces should probably access only team volumes. If individual user volumes must exist in a team namespace, think carefully about the implication of having a users credential inside a secret in that namespace and the RBAC required to protect it. If a vendor is generating the team or project namespace, ensure that they set the RBAC properly for secrets in that namespace

CHALLENGE: DATA SECURITY & SERVICE ACCOUNTS

- Data Scientists live in notebooks
- Notebooks exist in containers. These containers are launched by Jupyter Hub and use Istio to route a users browser request to the right container
- Container must therefore normally run as a service account
- There is a need for the container running as the user to perform many data accesses (reading a file) and Kubernetes API server operations (submitting a job) on behalf of the user
- The notebooks service account complicates things significantly:
  - Security model becomes more difficult to create (assigning RBACs to SA instead of user)
  - Must ensure that all the proper tokens for backends are available and mounted inside notebook
  - Must deal with various auditability concerns
- Tip: Use webhooks (admission controllers), Istio, and JupyterHub to ensure secrets with the proper credentials are created and mounted to the notebook container when it is created
- Tip: Be very careful with allow exec RBAC for notebook containers

MORE INFORMATION

Email: skyler.thomas@hpe.com

HPE Ezmeral ML Ops: https://www.hpe.com/us/en/software/machine-learning-operations.html
HPE Ezmeral Data Fabric: https://www.hpe.com/us/en/software/data-fabric.html