ML Ops Guide

April 2020

Hubert Duan, Cloud Solution Architect

Architecture

Automation

Azure DevOps

Build Release Pipelines

Data Orchestration

Data Factory

Data movement

Data

Experimentation

Azure Databricks

Feature engineering

Modeling

Model management and deployment

Azure ML services

Model lifecycle

Deploy to AKS

Model performance tracking

Power BI

Getting started

Azure provisioning

This workshop will assume services are organized within a resource group, so go ahead and create a new RG along with these services:

- Azure Data Factory
- Azure Data Lake Gen 2
  - Hierarchical namespace enabled
- Azure SQL DB
- Azure Databricks
  - Premium tier
- Azure Machine Learning
  - Enterprise edition
- Azure Kubernetes
  - At least 12 cores cluster needed
- Azure Key Vault

Also, this workshop will utilize an Azure DevOps project, so create one as well

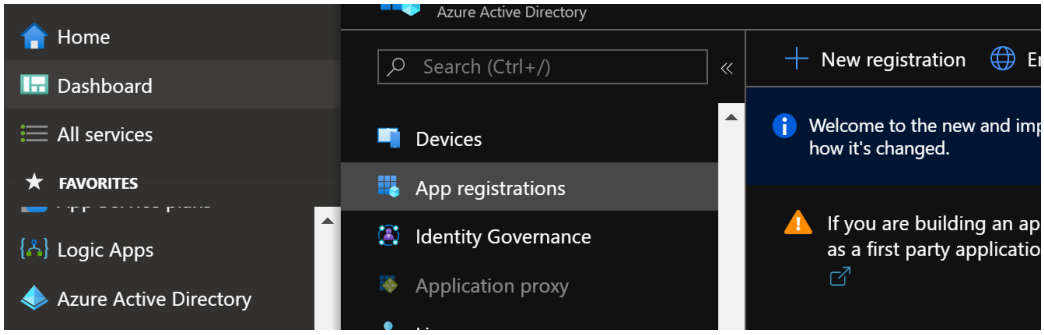
Getting started

Service Principal integration

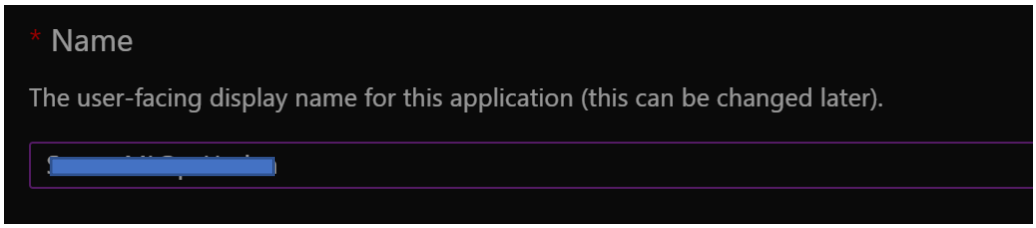
We will use a service principal for pipeline and service integration / automation

- Go to Azure Portal and click on

Azure Active Directory → App registration → new registration



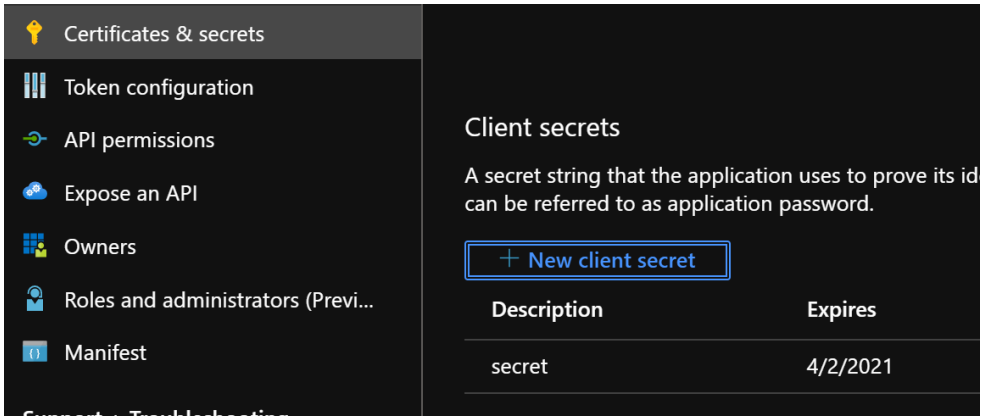
- Give it a name and register it



Getting started

Service Principal integration

- Copy the application ID, tenant ID as they will be used later
- Copy the subscription ID as well, you can find this by clicking on the resource group and going to Overview
- Go to certificates & secrets and click on + New client secret to create one. Save the secret as it will be used later as well

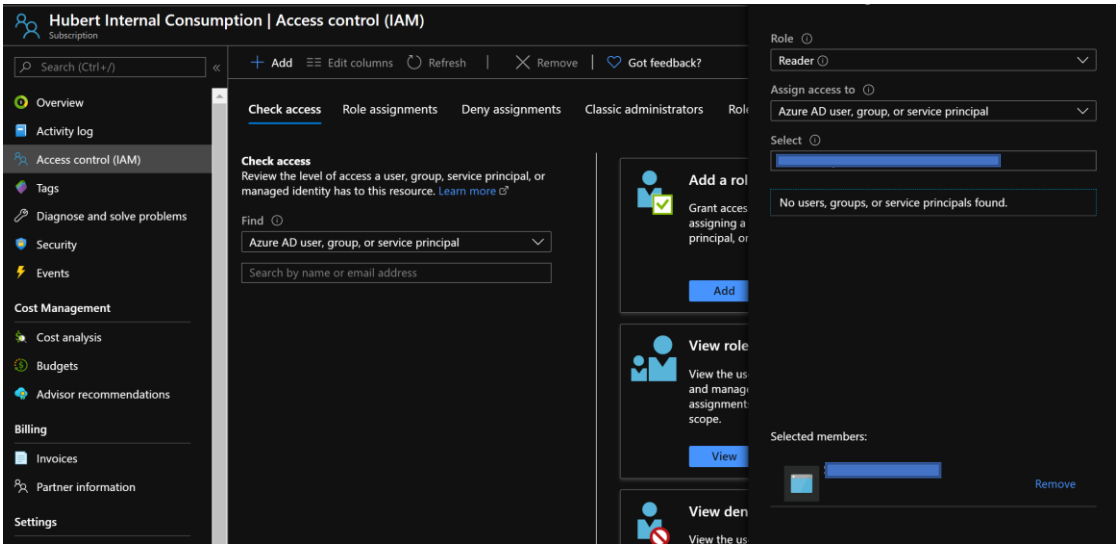


Getting started

Service Principal integration

This service principal will also be used for Azure DevOps pipelines through a service connection so will require read access to the subscription

- On Azure Portal, visit this subscription
- Under IAM (Access control) → Add role assignment
- Pick Reader and search the name of service principal

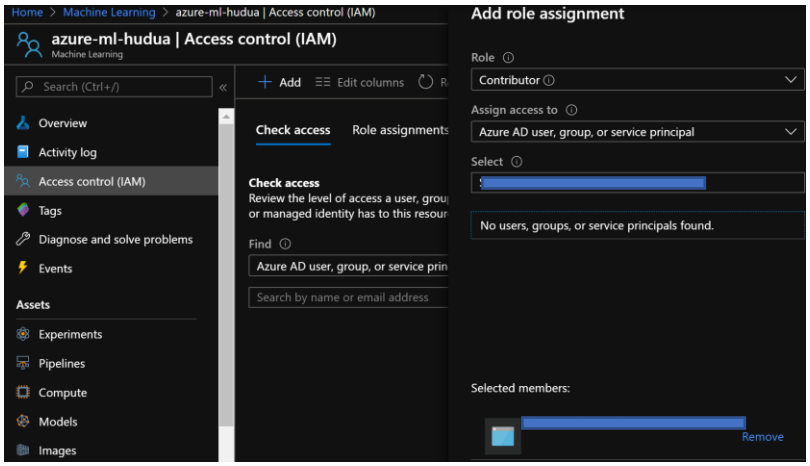


Getting started

Service Principal integration with AML

We will allow contributor access for the service principal with Azure Machine Learning

- Go to the AML service, that was just created, overview section
- Click on IAM (Access Control) → Add → role assignment
- Select and create role as contributor and you can search for the name of the service principal that was just created

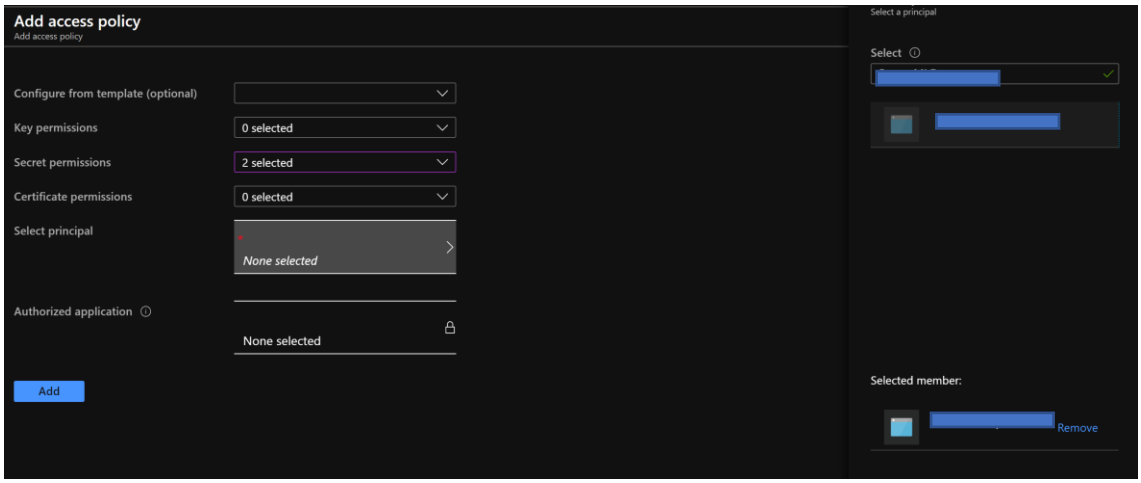


Getting started

Service Principal integration with Key Vault

We will grant get and list permission for the service principal with Azure Key Vault

- Go to Azure Key Vault, that was just created
- Click on Access policies → Add Access Policy
- Under Secret permissions, select Get, List
- Click on select principal and search for the service principal name

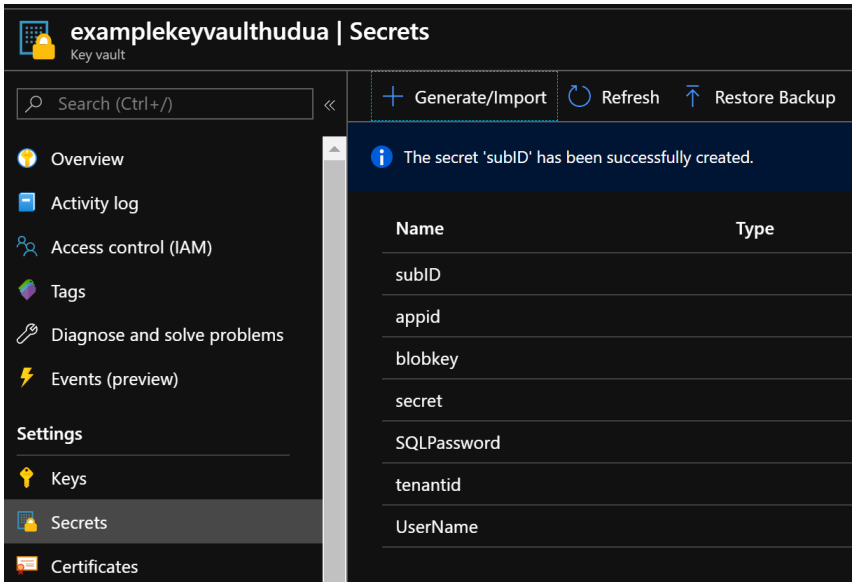


Getting started

Service Principal integration with Key Vault

Now, add the service principal connection information as Key Vault secrets

- Go to Azure Key Vault, that was just created
- Click on Secrets → Generate / Import
- Add in the subscription ID, tenant ID, application ID, and secret, as well as username and password of the SQL DB

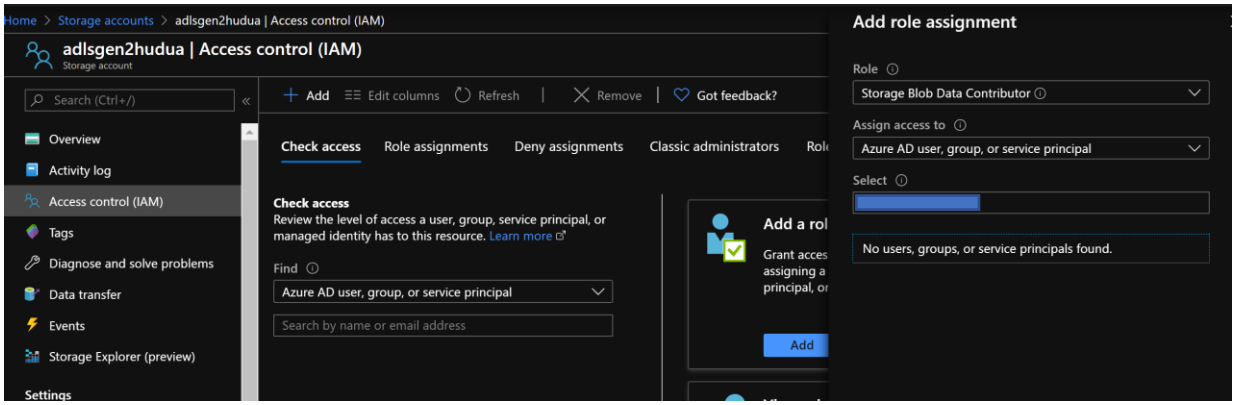


Getting started

Service Principal integration with ADLS Gen 2

We will provide Blob Data Contributor access for ADLS Gen 2 with the service principal so Databricks can be mounted on ADLS Gen 2

- Go to ADLS Gen 2, that was just created
- Click on IAM (Access control) → Add role assignment
- Select Storage Blob Data Contributor and search for the service principal name



Getting started

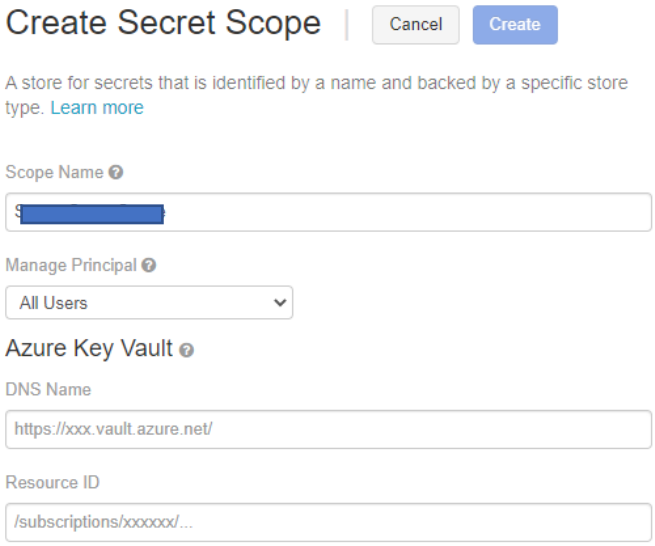
Azure Key Vault integration with Databricks

We will now integration Azure Key Vault with Azure Databricks to manage and access secrets

- Access this site, based on the region Databricks is provisioned in

https://<region>.azuredatabricks.net/#secrets/createScope

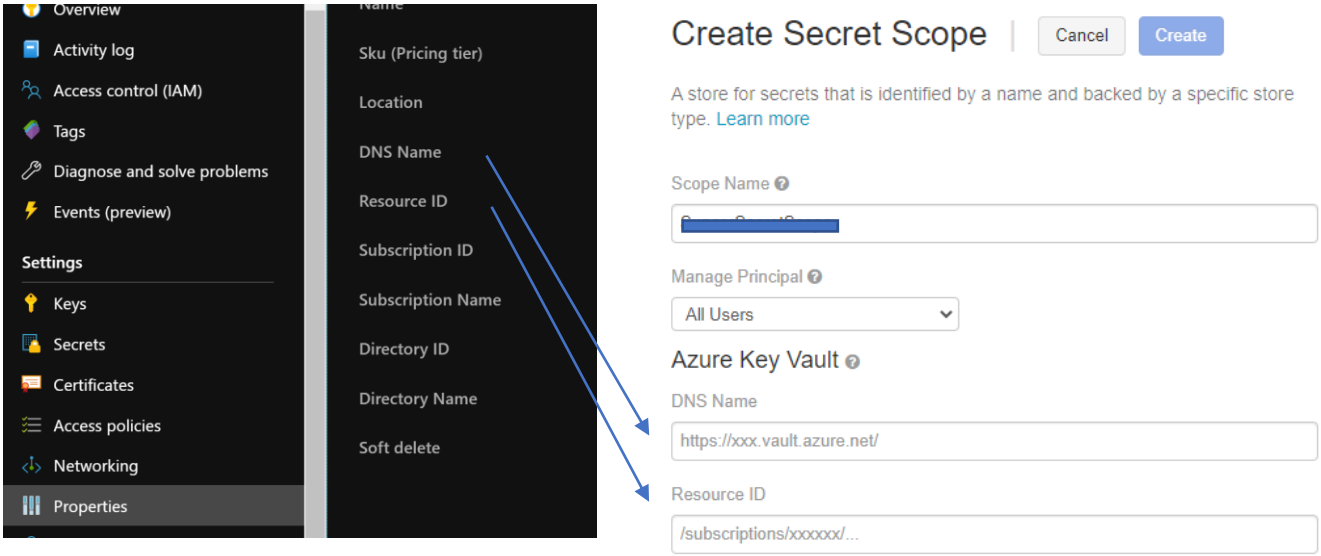
- Give it a scope name such as SuncorSecretScope
- Select All Users



Getting started

Azure Key Vault integration with Databricks

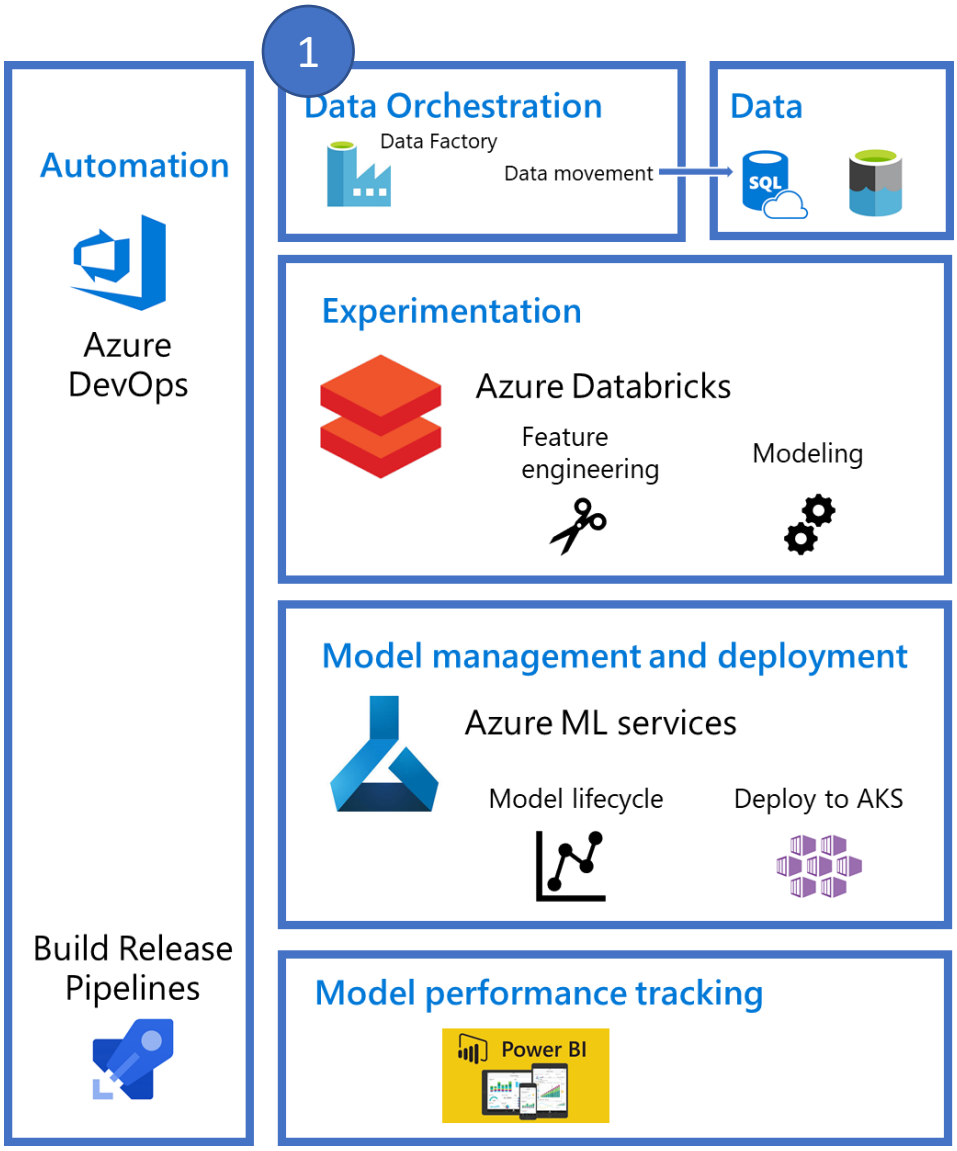
- Use another tab, and visit Azure Portal and go to the Azure Key Vault service
- Under properties, you can find DNS name and resource ID. Copy them into the Create Secret Scope Databricks page, and create the link



1. Data Ingestion

Azure Data Factory

Now we are ready to work on the architecture components, starting with data movement



1. Data Ingestion

Azure Data Factory

Bonus: depending on the model and data sources required, set up an Azure Data Factory pipeline, e.g. on-premise to Azure, ADLS Gen 2 transformation, ADLS Gen 2 to SQL DB, etc.

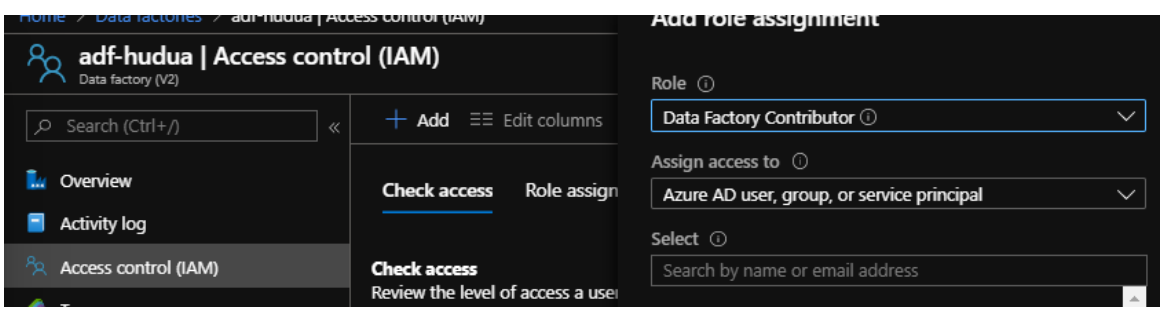
The end outcome for this section 1) is an ADF pipeline that is published

1. Data Ingestion

Azure Data Factory

Bonus: You will need to provide create Run permission for the service principal for Data Factory

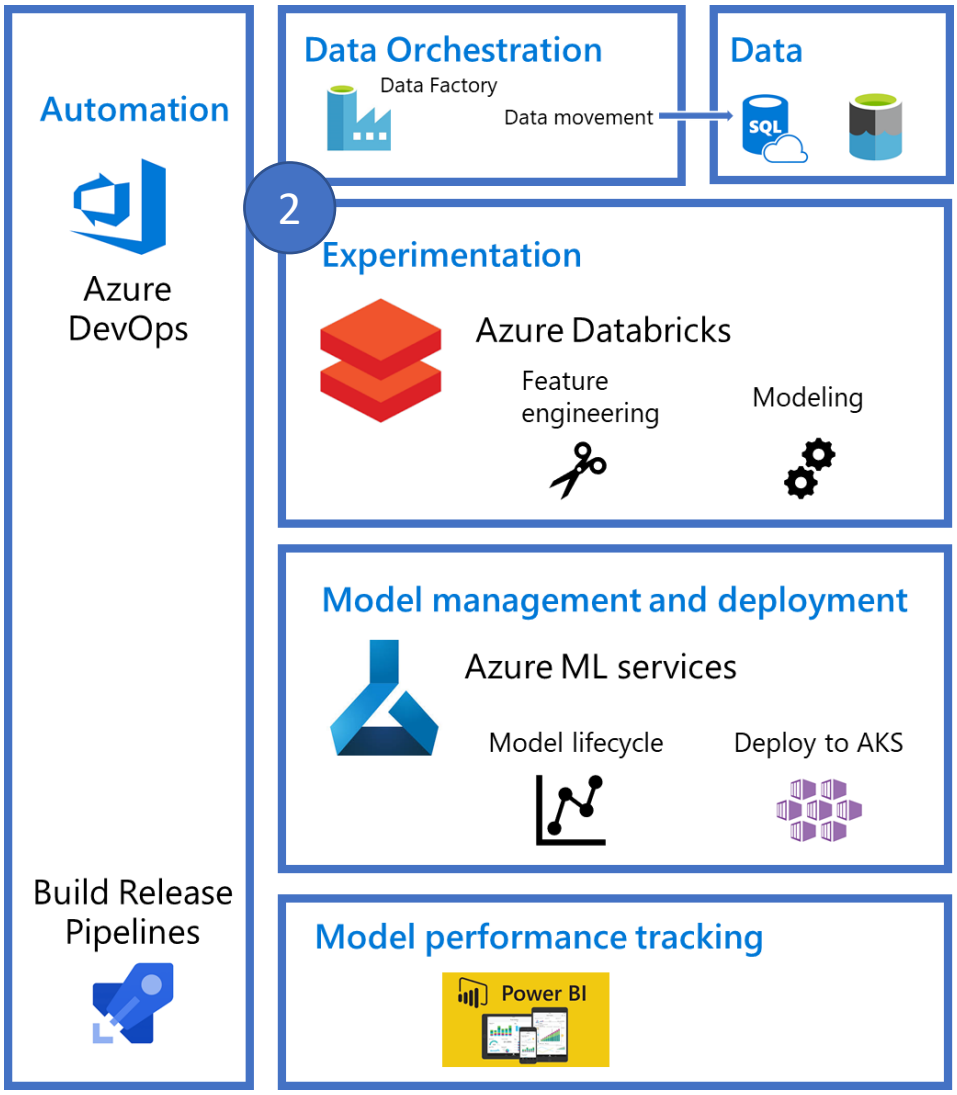
- Visit Azure Portal and go to the Data Factory service
- Click on IAM (Access Control) → Add → Add role assignment
- Give it role Data Factory Contributor, search for the service principal name, and save



2. Experimentation

Azure Databricks

Now we will work on the Azure Databricks Experimentation

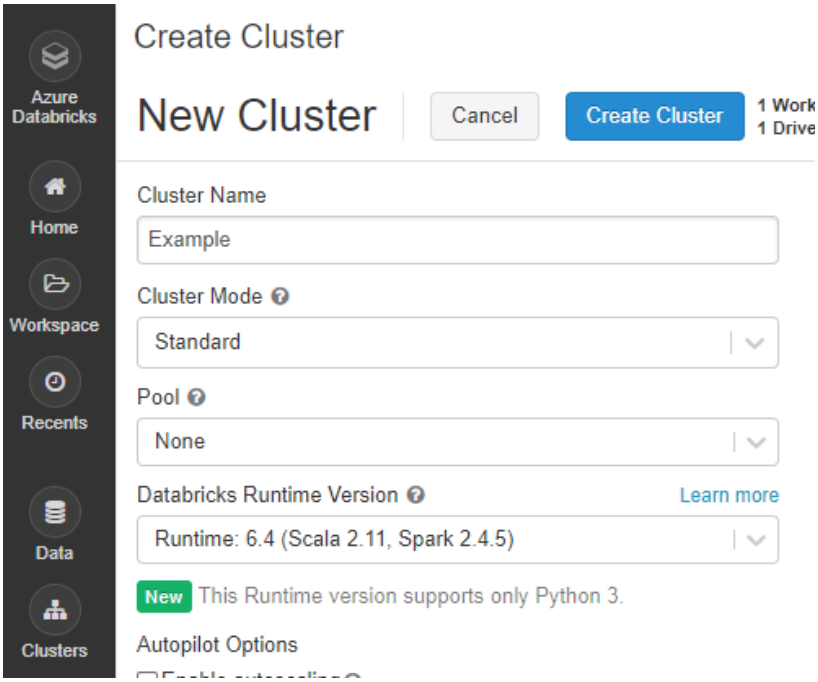


2. Experimentation

Azure Databricks

We will perform work on Azure Databricks

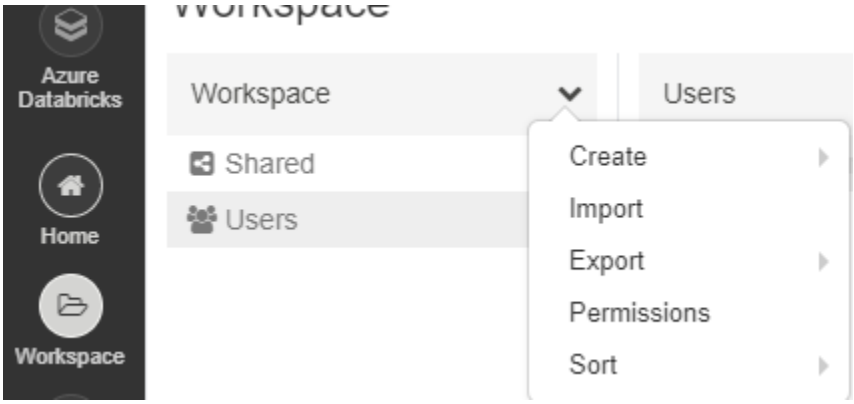
- Go to the Databricks workspace and create a cluster
- You can provide it a name and keep default settings



2. Experimentation

Azure Databricks

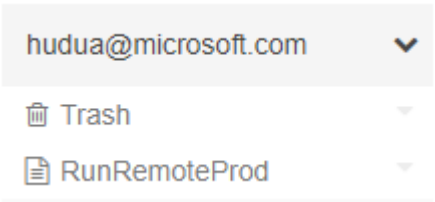
- Go to the Databricks workspace and click on the workspace icon
- Click on the drop down and select Import
- Upload the .dbc file to the workspace

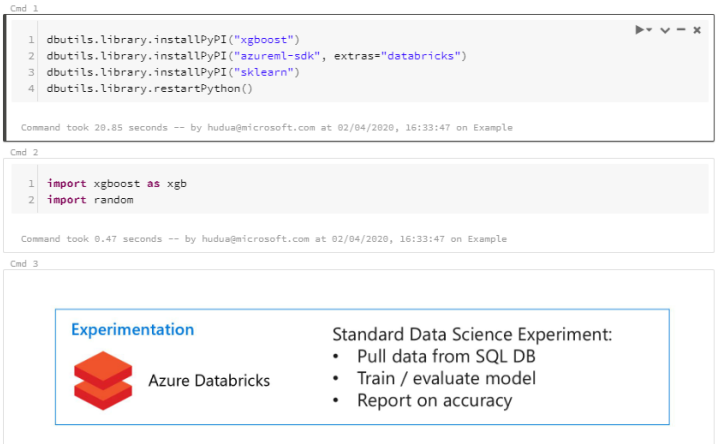


2. Experimentation

Azure Databricks

- Access this workspace notebook. The assumption is that data for modeling sits within an Azure SQL DB





- The code performs the following
  - Installs the right libraries
  - Pulls in data from SQL DB
  - Performs machine learning training and evaluation
- Then scroll down to Access Workspace, where the code using AML to log experiment results and register model

2. Experimentation

Azure Databricks

- Change the name of the resource group
- Ensure the secret names and scope are correct

Access workspace

Cmd 15

```
tenant_id = dbutils.secrets.get(scope = '[redacted]', key = 'tenantid')
subscription_id = dbutils.secrets.get(scope = [redacted], key = 'subID')
app_id = dbutils.secrets.get(scope = '[redacted], key = 'appid')
secret = dbutils.secrets.get(scope = '[redacted], key = 'secret')
rg = [redacted]"
```

Command took 2.82 seconds -- by hudua@microsoft.com at 02/04/2020, 16:33:47 on Example

Cmd 16

```
svc_pr =
ServicePrincipalAuthentication(tenant_id=tenant_id,service_principal_id=app_id,service_principal_p
assword=secret)
ws = Workspace(
        workspace_name="azure-ml-hudua",
        subscription_id =  subscription_id,
        resource_group = rg,
        auth = svc_pr
    )
```

Command took 0.71 seconds -- by hudua@microsoft.com at 02/04/2020, 16:33:47 on Example

- Attempt to run this notebook with Run All after assigning to the cluster

RunRemoteProd (Python)

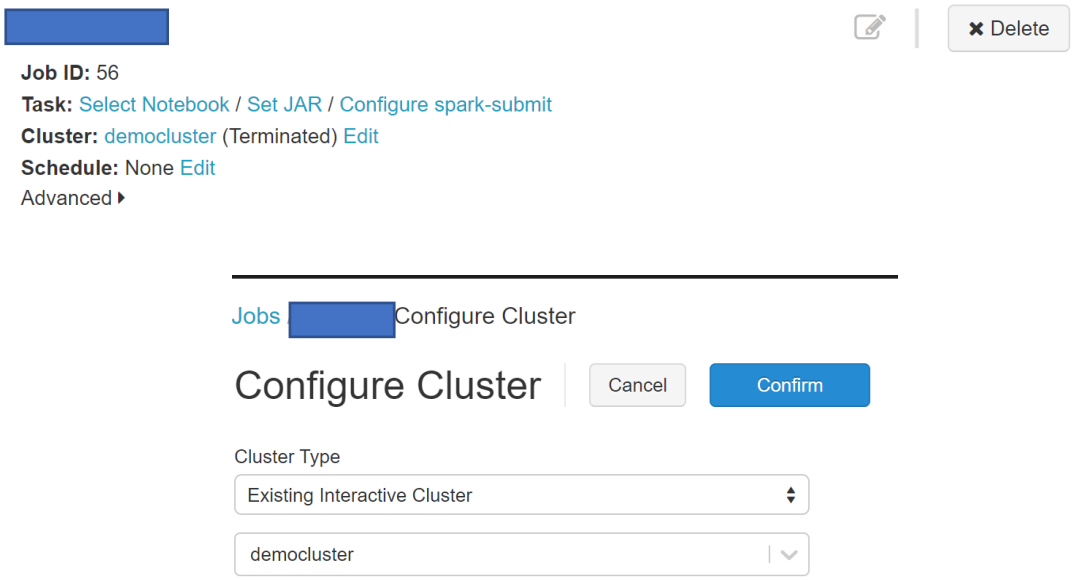
Example | File | View: Code | Permissions | Run All | Clear

2. Experimentation

Azure Databricks

Now you can publish this notebook as a job

- Go to Job → Create job and enter a name
- Select the notebook that was worked on
- Under cluster, click on edit and select Cluster Type with existing
- Pick the provisioned cluster one

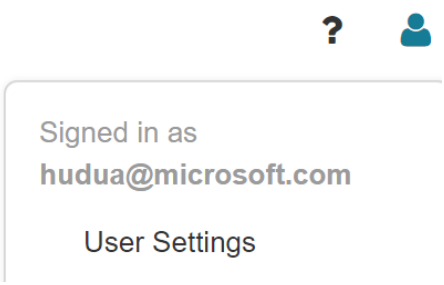


2. Experimentation

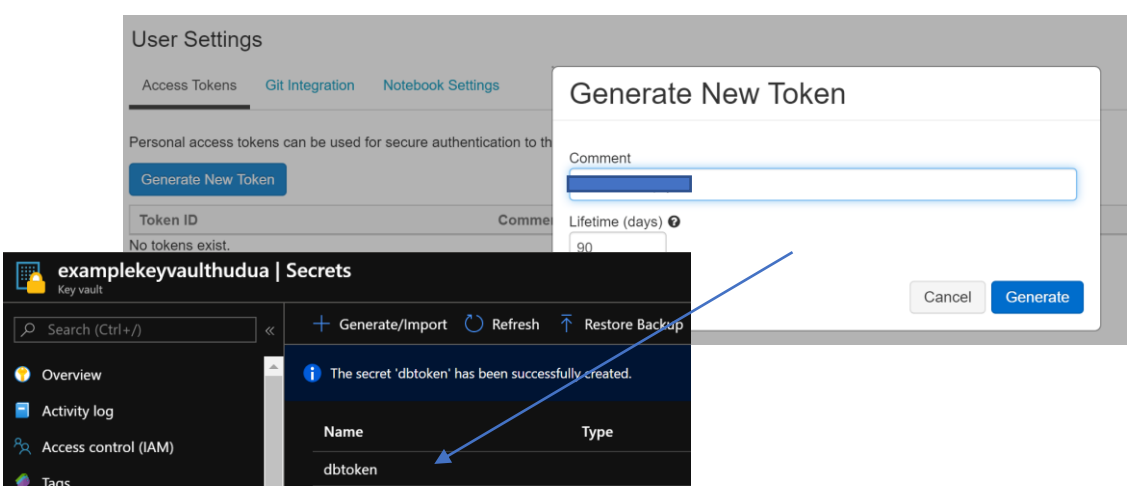
Azure Databricks

Finally get a Databricks token so this job can be triggered through Databricks Job API

- Go to User Icon → User Settings



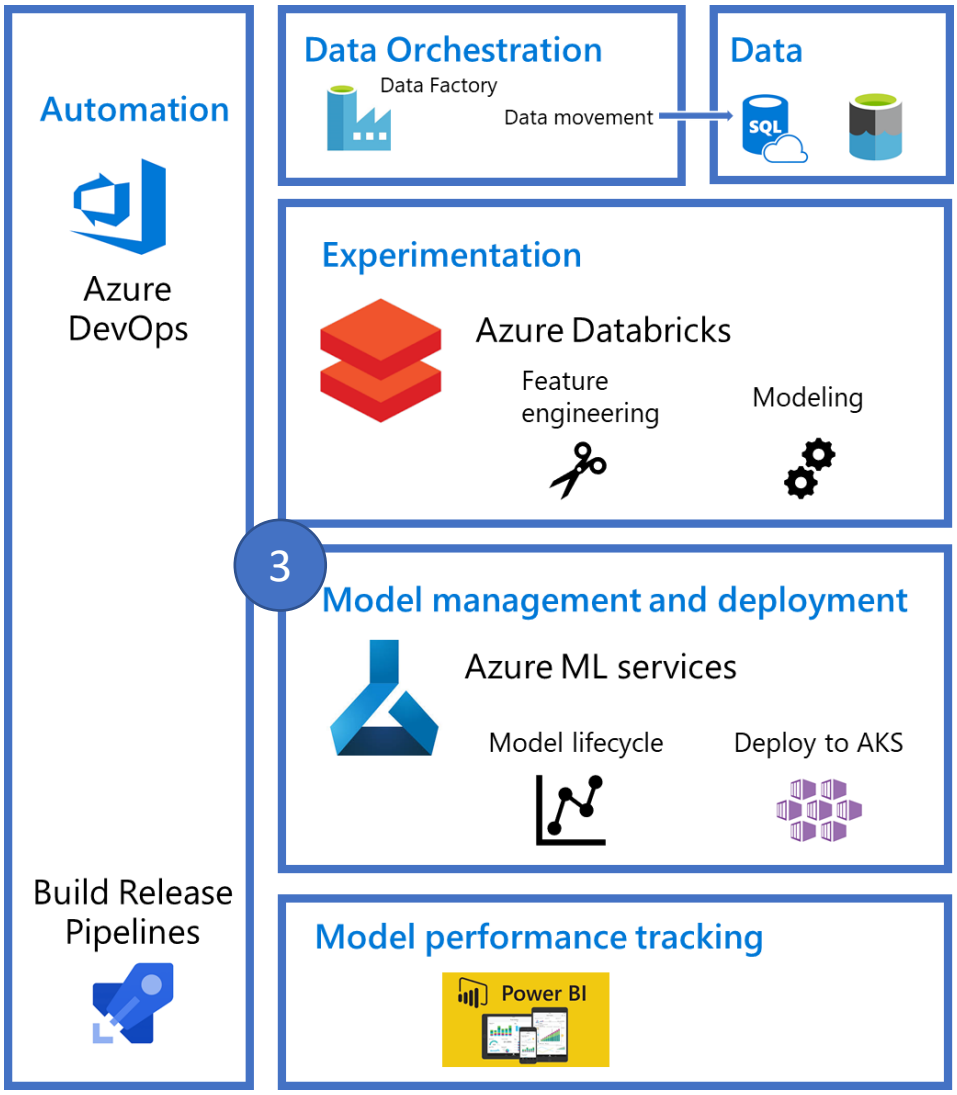
- Click on Generate New Token and save it to the Key Vault secret vault



3. Model

Azure Machine Learning

Here is how Azure Machine Learning service works

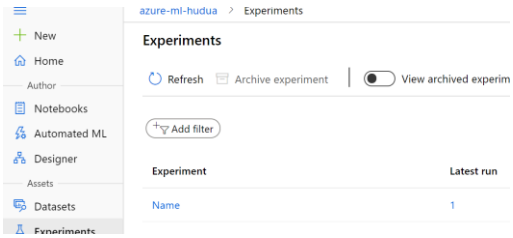


3. Model

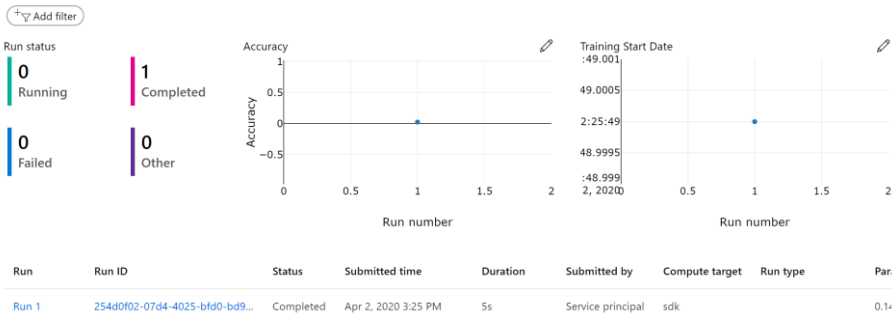
Azure Machine Learning

Under Azure Machine Learning service, you can link to https://ml.azure.com

- Under experiment, you should be able to see the name and you can see the experimentation tracking metrics



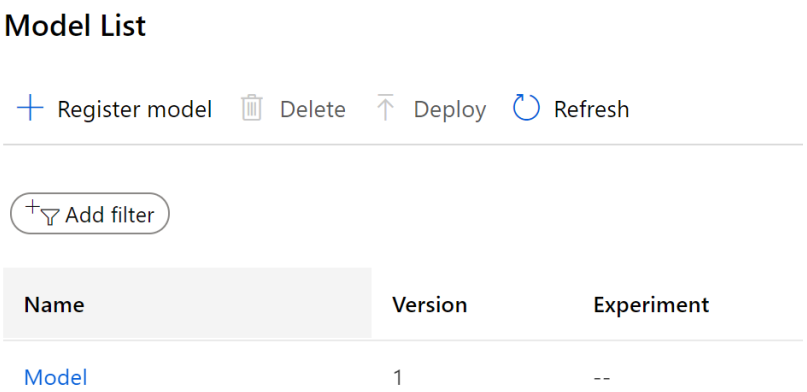
- You can make edits to the graphs and see the historical runs



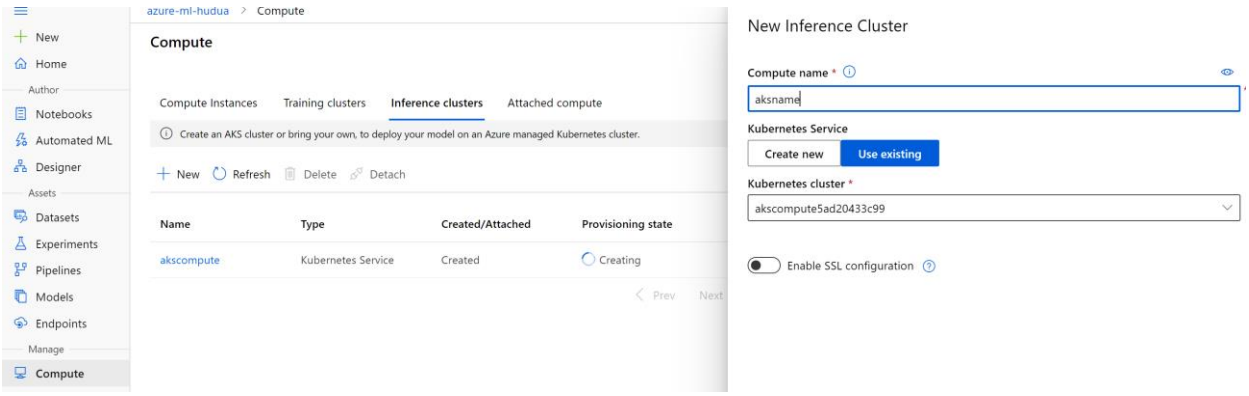
3. Model

Azure Machine Learning

- Under model, there is a registry of the model that is uploaded



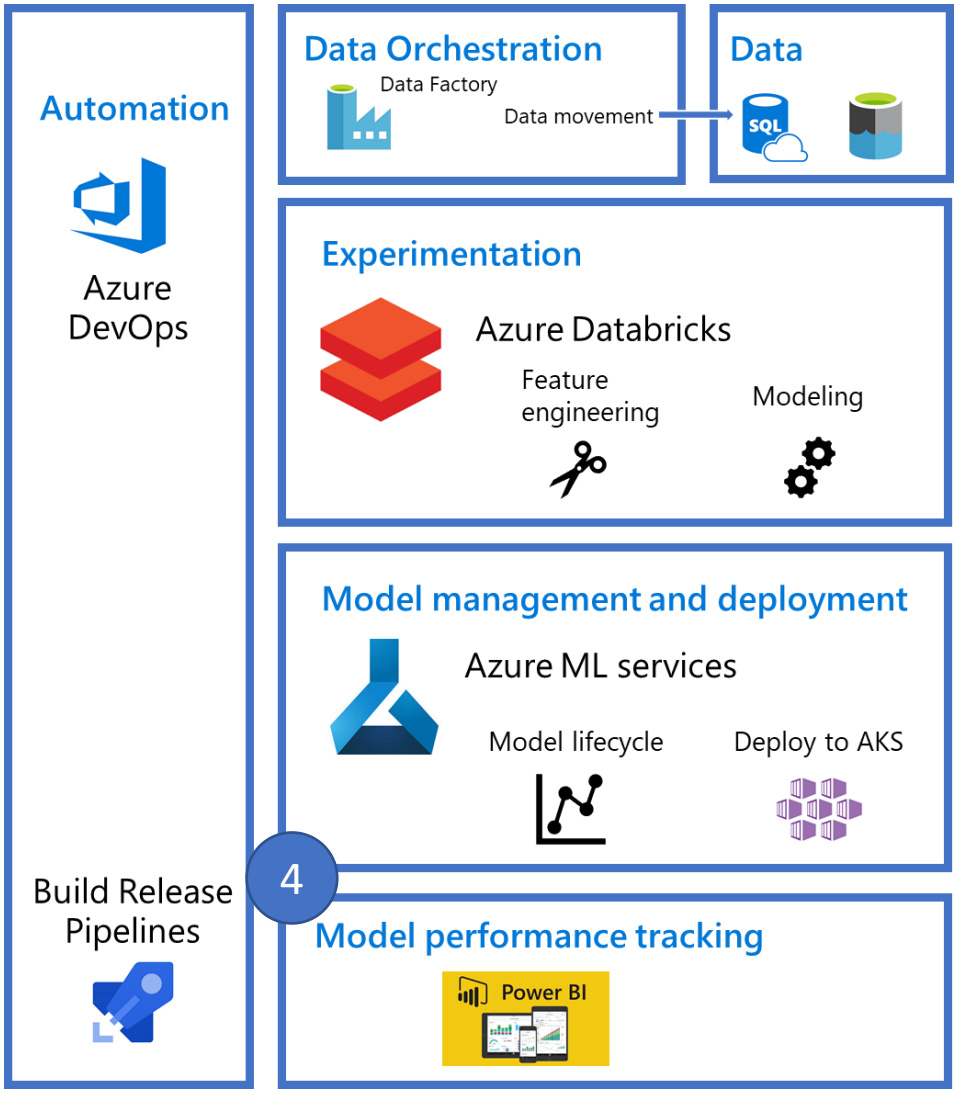
- Then under compute, you can select Inference cluster and attach the provisioned AKS cluster for model deployment



4. Reporting

Power BI integration

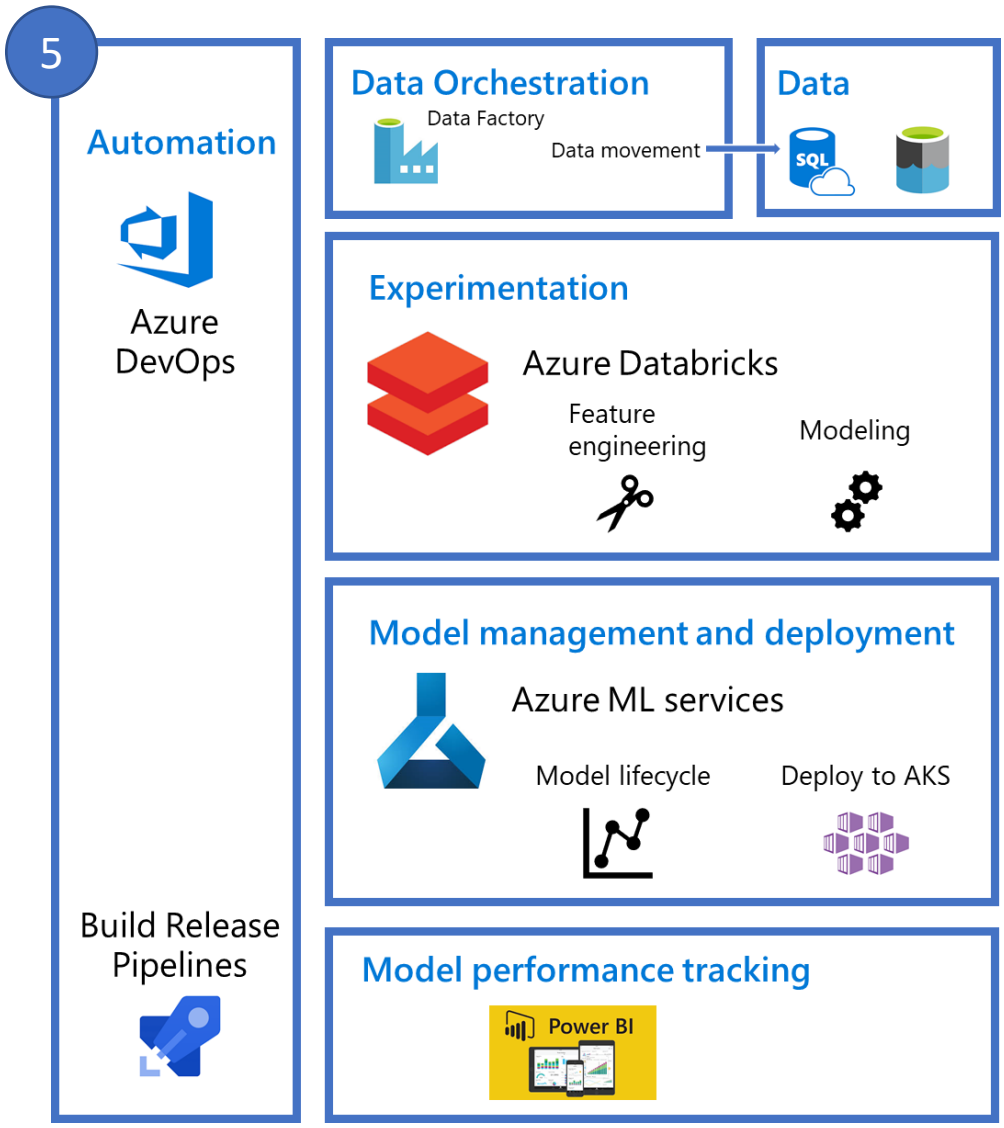
Bonus exercise: use the Python SDK to extract out AML metrics into PBI for reporting purposes



5. ML Ops

Azure DevOps integration

We will focus on Azure DevOps including the repository as well as build, release pipelines

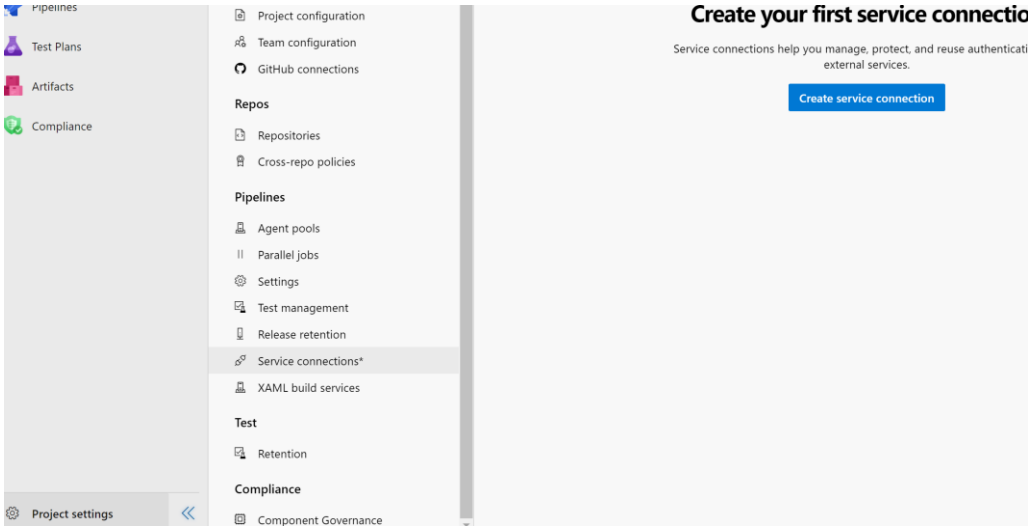


5. ML Ops

Azure DevOps integration

With the newly created ADO project…

- Go to Project settings → service connections → Create service connection

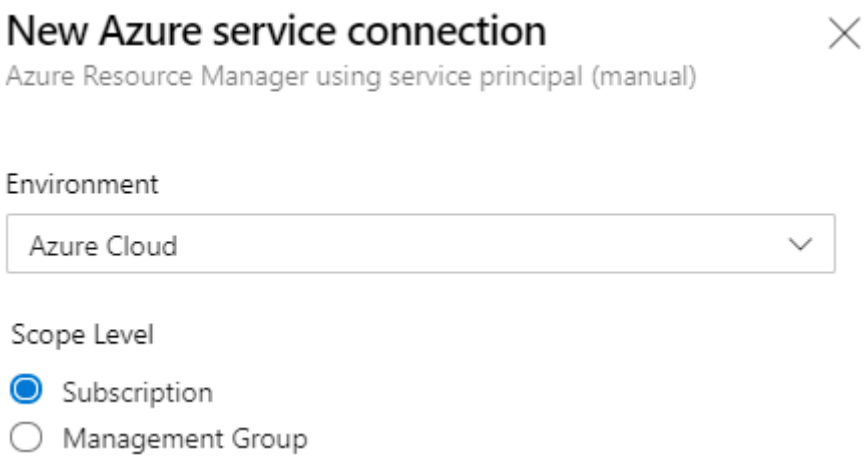


- Pick Azure Resource Manager → Service principal (manual)

5. ML Ops

Azure DevOps integration

- Enter the subscription ID, subscription Name, as well as the service principal Id (app ID), secret, and tenant ID
- These can all be found in the Key Vault or the safe locations of IDs after creating the service principal



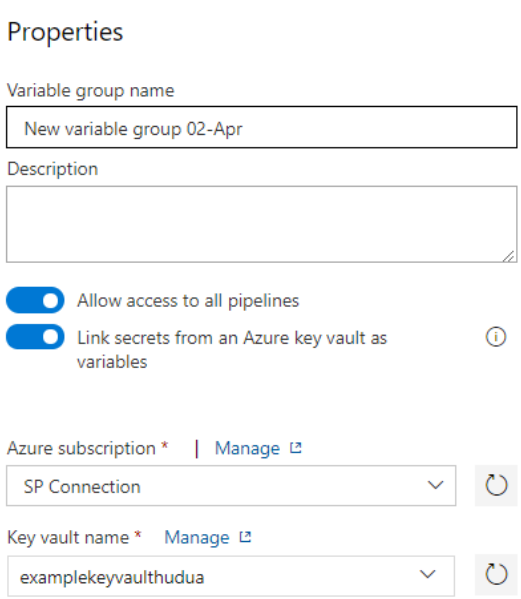
- Click on verify and then enter a service connection name

5. ML Ops

Azure DevOps integration

Now we will integrate Azure Key Vault and list the secrets as variable group in ADO project

- Highlight pipeline and select Library → Select variable group and + Variable group
- Give it a name and click on link secrets
- Select the new SP connection and pick the key vault name



5. ML Ops

Azure DevOps integration

- Click on Add variables and select the secrets to add in

Choose secrets

Choose secrets to be included in this variable group

Sel...	Secret name	Content type	Status	Expiration date
☑	appid		Enabled	Never
☑	blobkey		Enabled	Never
☑	dbtoken		Enabled	Never
☑	secret		Enabled	Never
☑	SQLPassword		Enabled	Never
☑	subID		Enabled	Never
☑	tenantid		Enabled	Never
☑	UserName		Enabled	Never

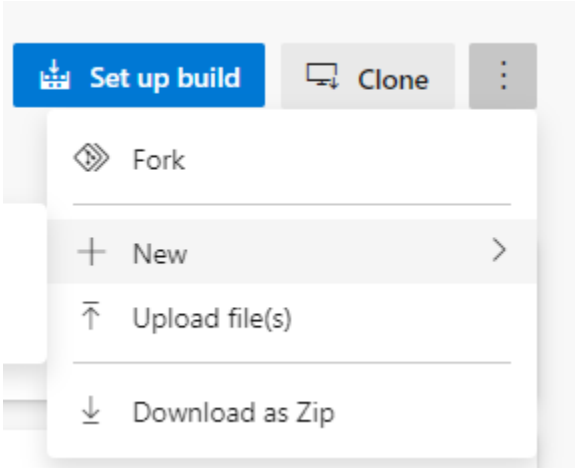
Ok Cancel

5. ML Ops

Azure DevOps integration

We will bring in a sample code repository from the shared files

- Go to Repos → Initialize with a README
- Click on options and upload files



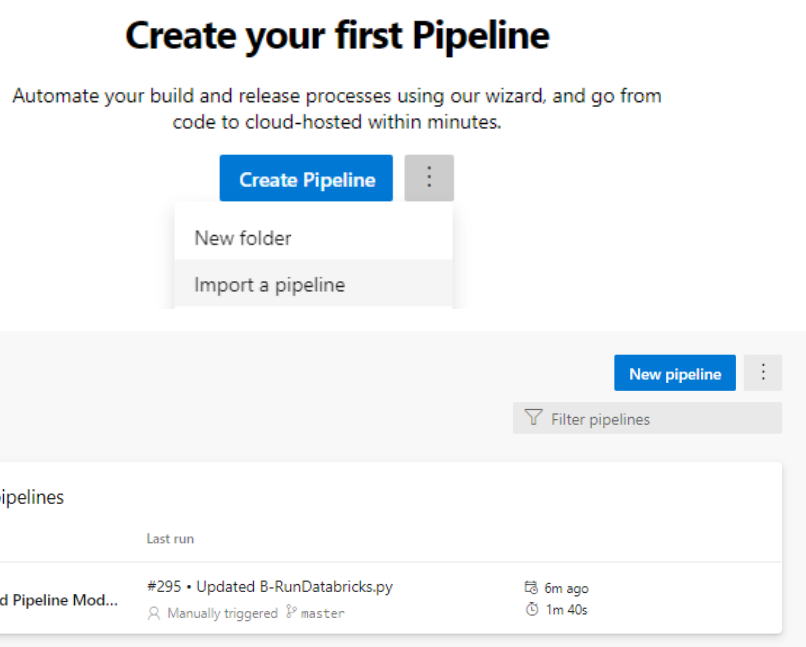
- Upload all the local files in the DevOps repo folder to this directory

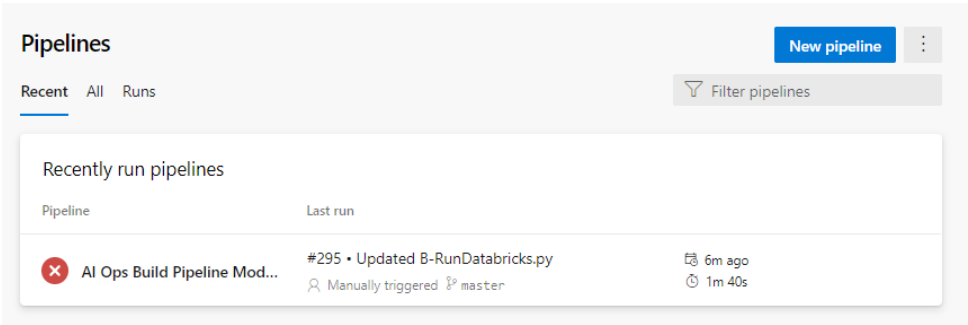
5. ML Ops

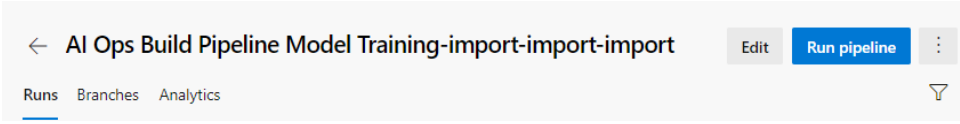
Azure DevOps integration

We will now upload the build pipeline template

- Go to Pipelines → Click on import new pipeline
- Upload the json pipeline file
- Click on the pipeline and then edit





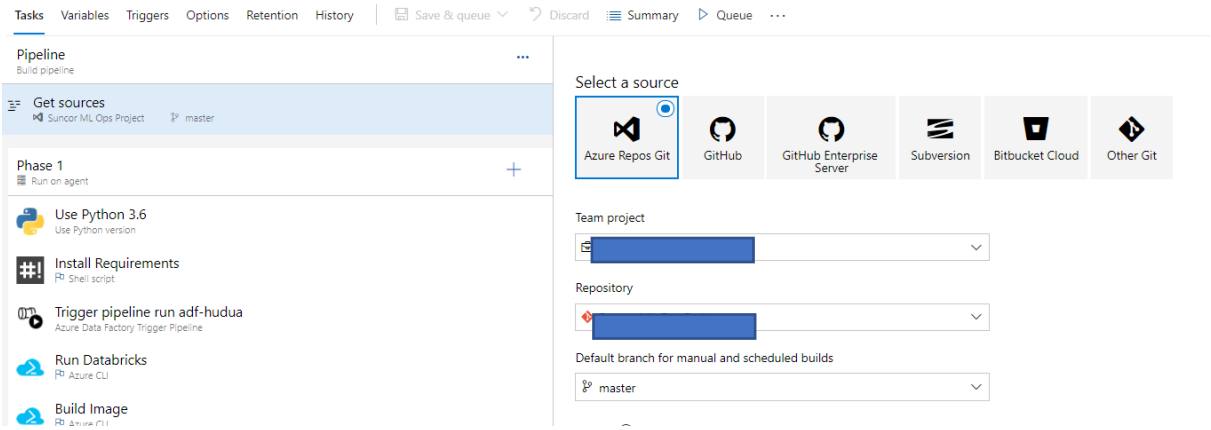


5. ML Ops

Azure DevOps integration

We will config get sources to pull from the repository with the uploaded files

- Go to Get sources → Azure Repos Git
- Select the proper team project and repository

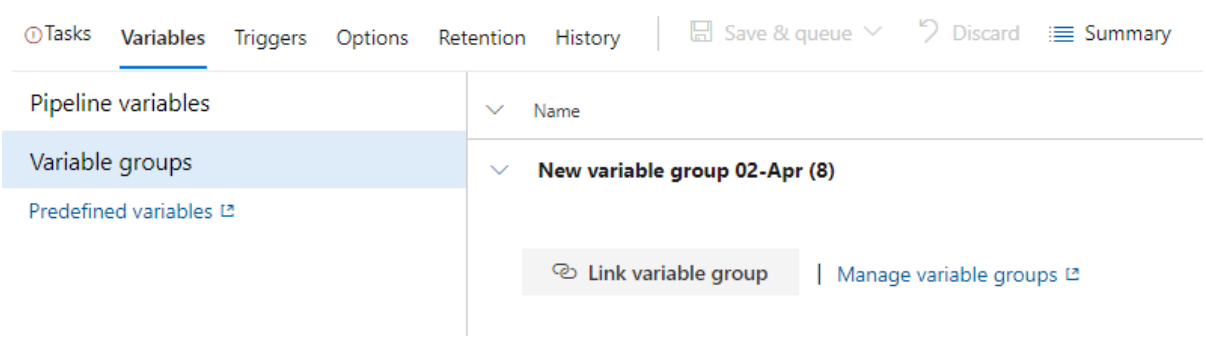


5. ML Ops

Azure DevOps integration

We will link the variable group (of Azure Key Vault secrets to this pipeline)

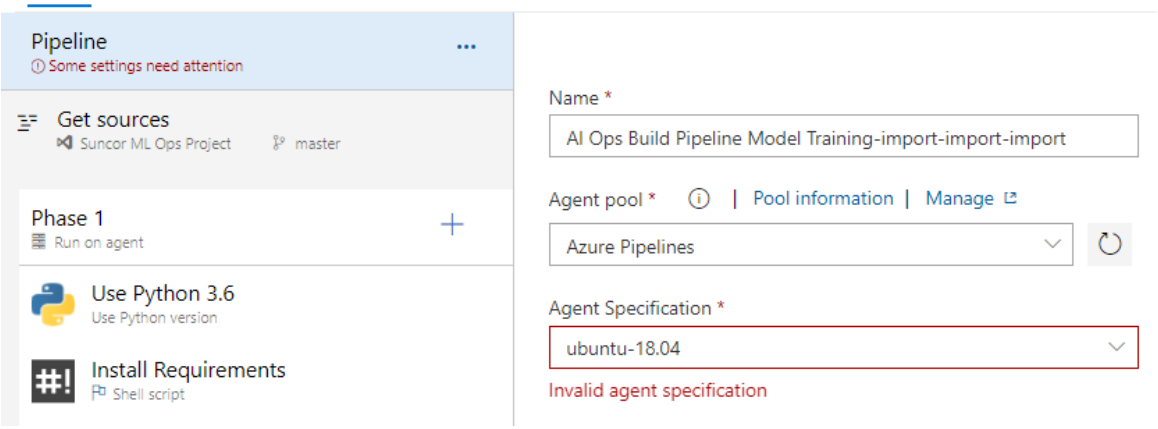
- Go to Variables → Variable groups
- Click on Link variable group and select the one created



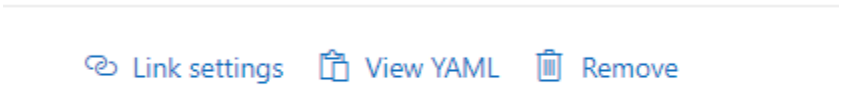
5. ML Ops

Azure DevOps integration

- Go back to tasks
- Select Azure Pipelines and ubuntu-18.04 as the agent pool and specifications



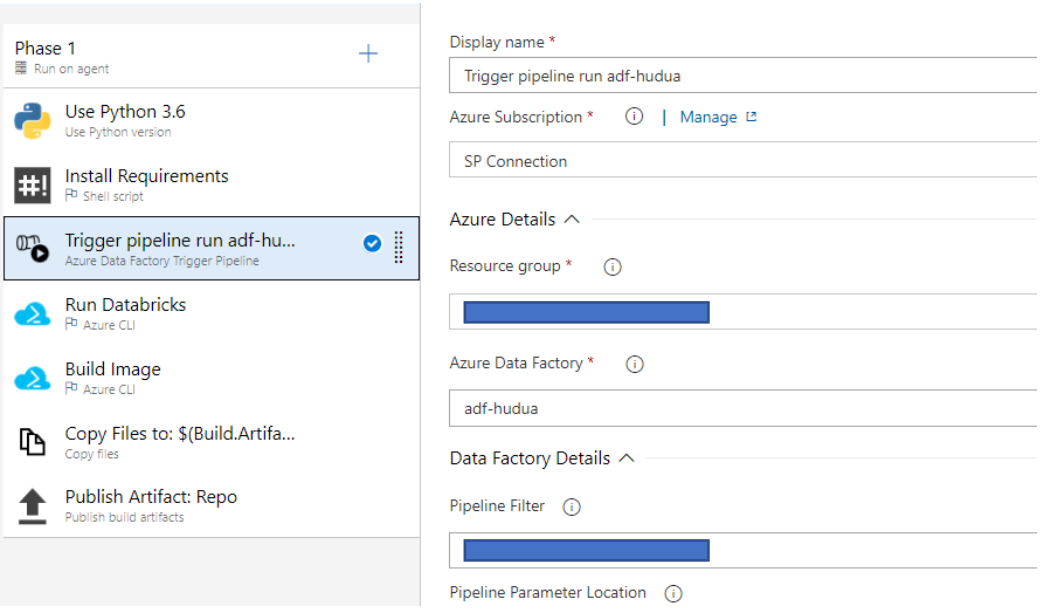
- (Bonus) If you completed Section 1) Data Movement, you can keep Trigger pipeline run task; otherwise, you can delete it



5. ML Ops

Azure DevOps integration

- (Bonus) For the Trigger pipeline task, select the service connection you created in ADO
- Pick the right resource group and enter the Azure Data Factory name
- Also enter the right pipeline filter (which is the pipeline name) as well

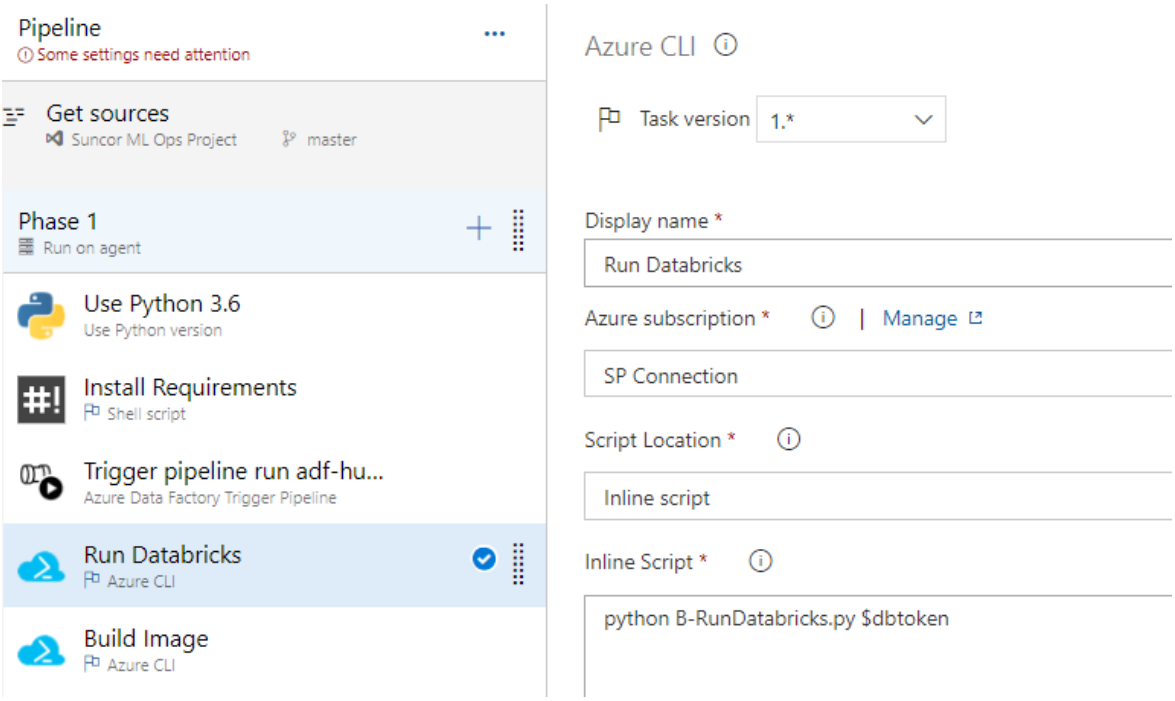


5. ML Ops

Azure DevOps integration

For the Run Databricks task:

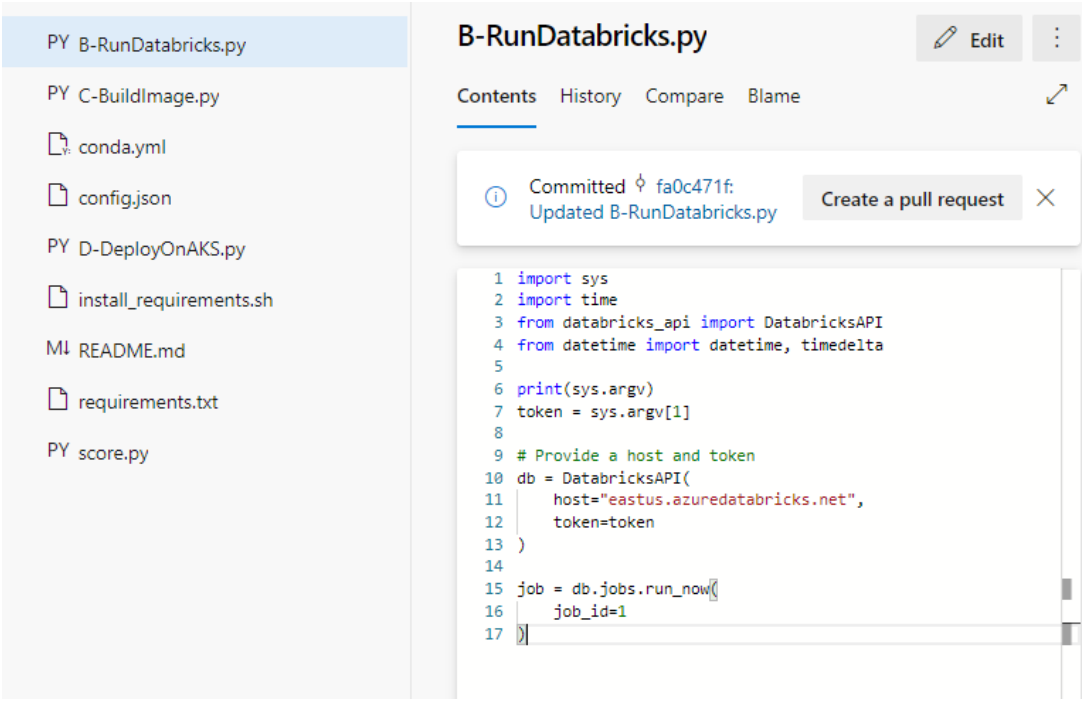
- Ensure the service connection created is selected and verify the databricks token variable is correct, from the variable group



5. ML Ops

Azure DevOps integration

- Also go to Repos and click on B-RunDatabricks.py
- Ensure the job_id is entered correctly; if not, you can click on Edit and make the change, followed by clicking on Commit

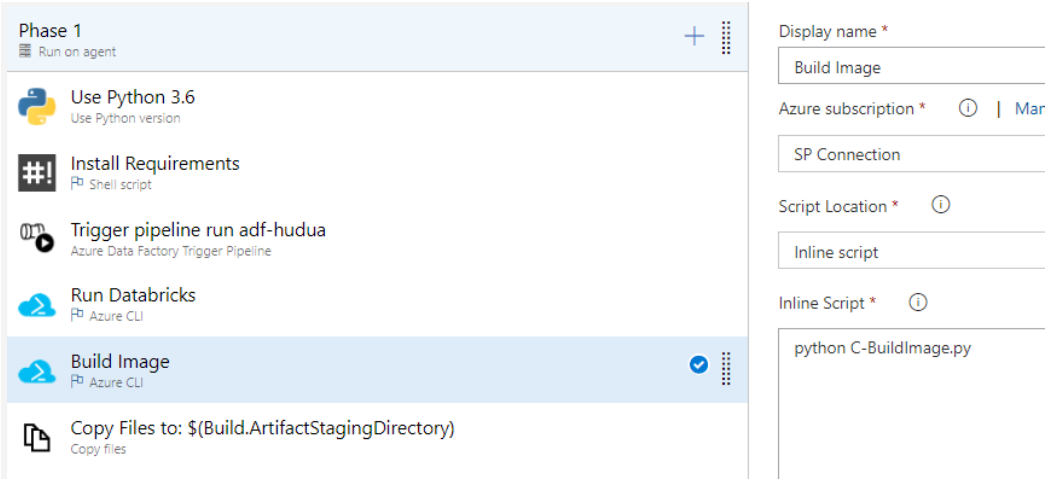


5. ML Ops

Azure DevOps integration

For the Build Image task:

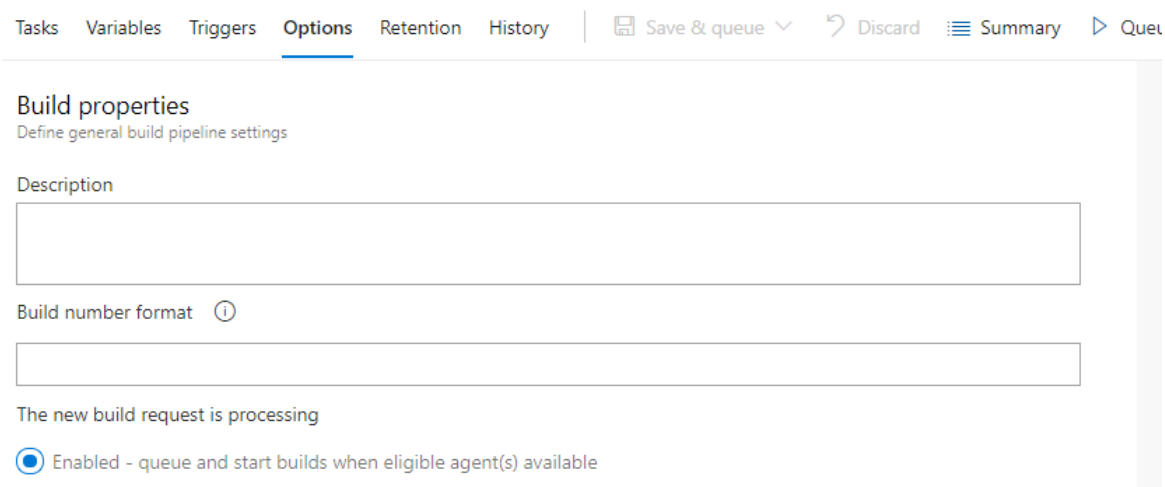
- Ensure the service connection created is selected



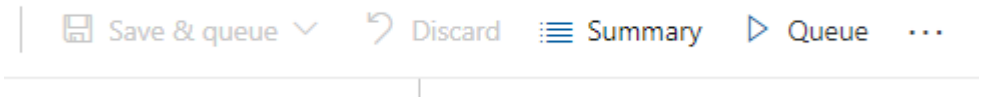
5. ML Ops

Azure DevOps integration

- Finally, go to Options and ensure “Enabled” build request is selected



- As always, make sure you save the pipeline consistently
- Finally, click on Queue to run your first Model Training (AKA Build) pipeline!

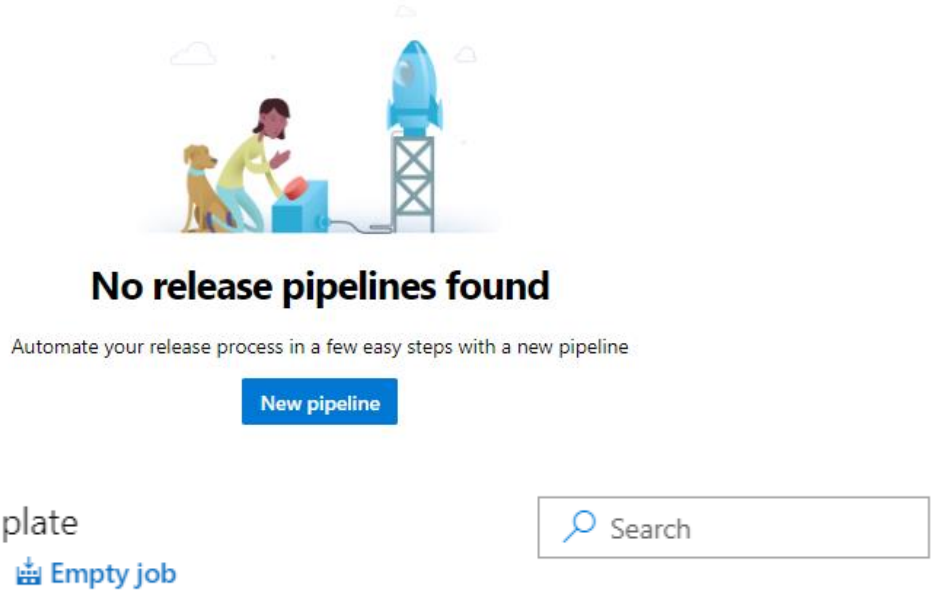


5. ML Ops

Azure DevOps integration

We will now upload the release pipeline template

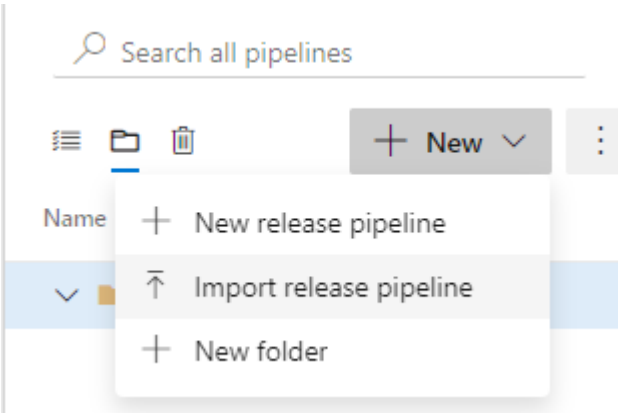
- First you have to create a new release pipeline (to be able to import another one)
- Click on Pipelines → Releases
- Click on New pipeline → Start with an empty job → Save



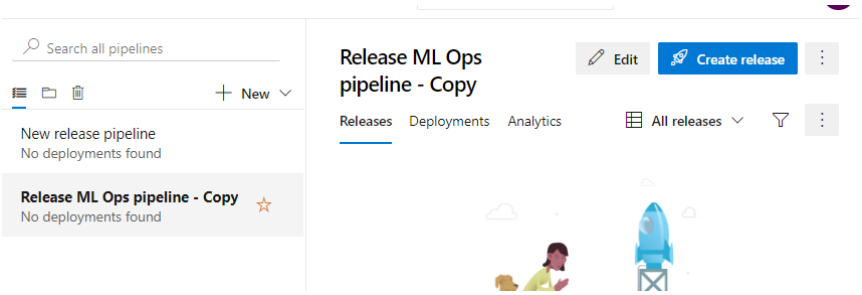
5. ML Ops

Azure DevOps integration

- Then click on Pipelines → Releases again
- Select + New and import new release pipeline, and upload the release pipeline JSON file



- Then select the uploaded release pipeline and click on Edit

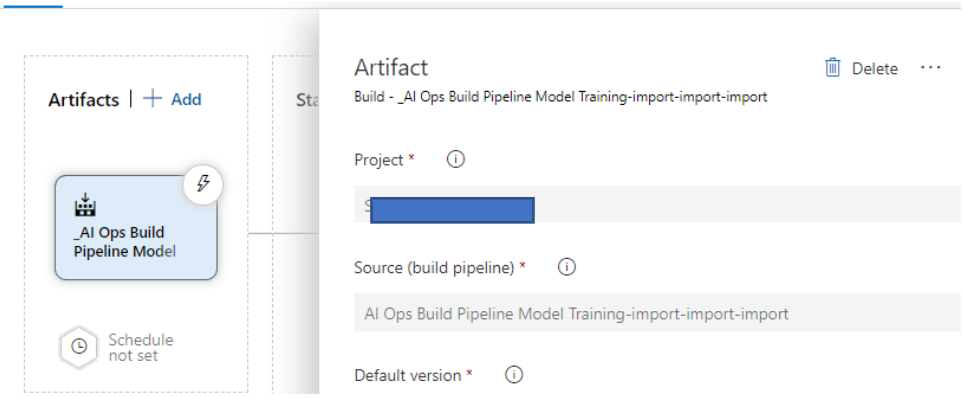


5. ML Ops

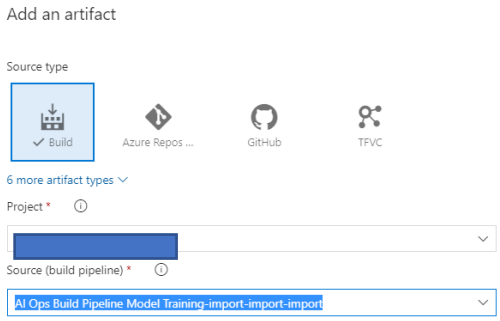
Azure DevOps integration

We will config the release pipeline artifacts and tasks:

- Click on the existing artifact and delete it



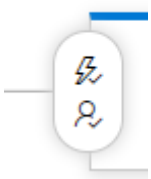
- Then click on Add an artifact and select the build source that was just completed

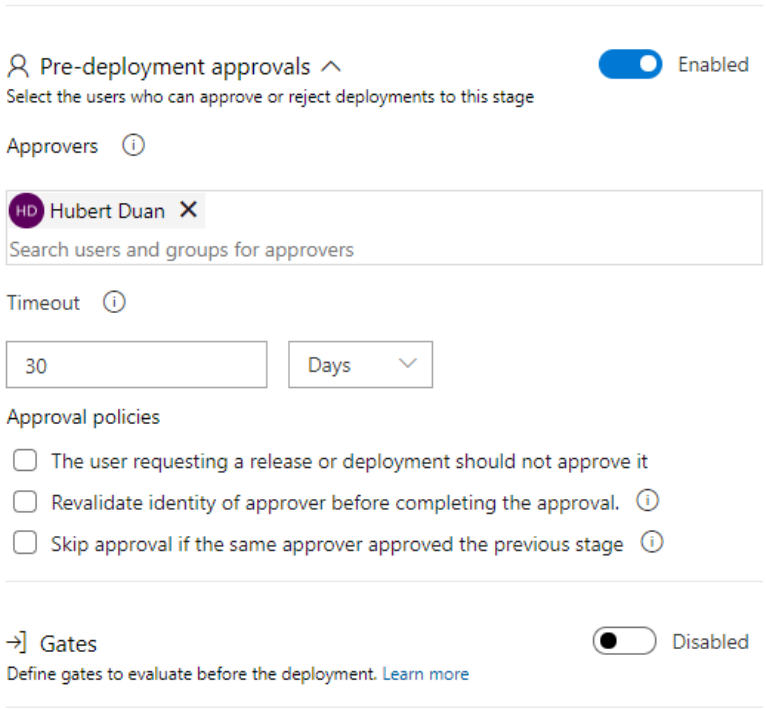


5. ML Ops

Azure DevOps integration

- Click on the icons before Deploy to AKS
- Select a pre-deployment approver and disable gate (so there's no delay in deployment)

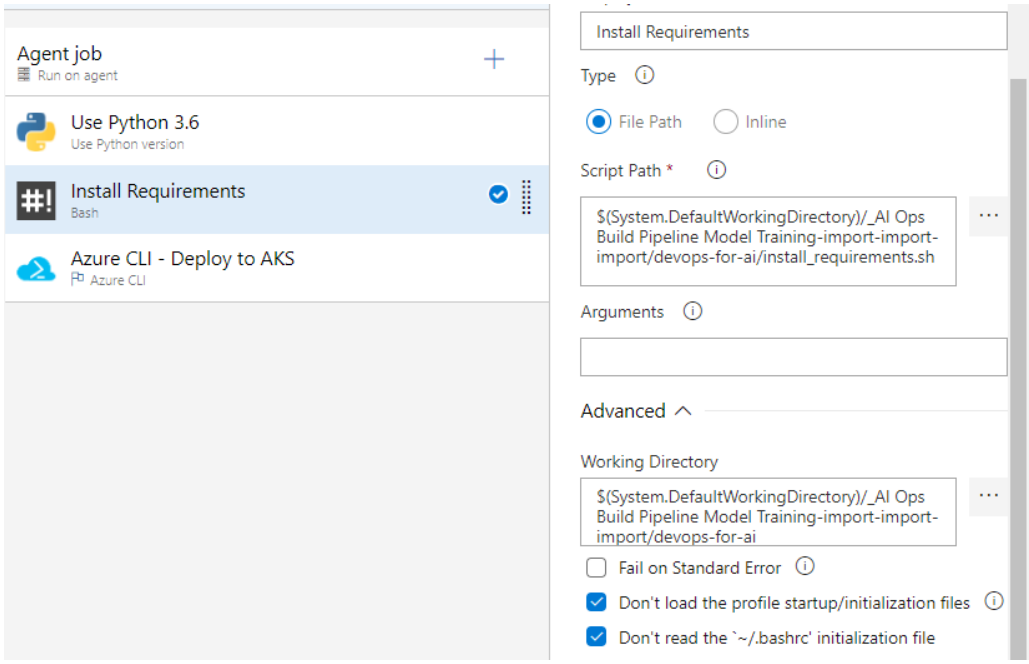




5. ML Ops

Azure DevOps integration

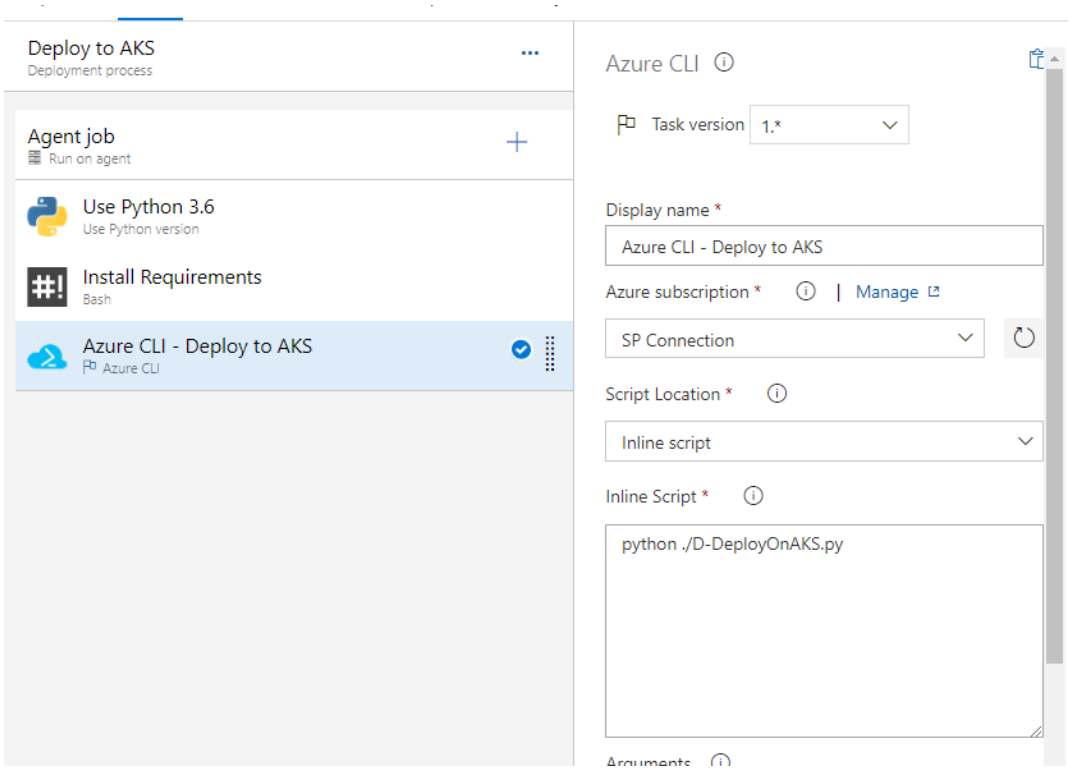
- Click on the Deploy to AKS tasks
- Select Install Requirements and browse to the right linked artifacts (under devops-for-ai) fo
  - Script path: install_requirements.sh
  - Advanced → working directory under devops-for-ai



5. ML Ops

Azure DevOps integration

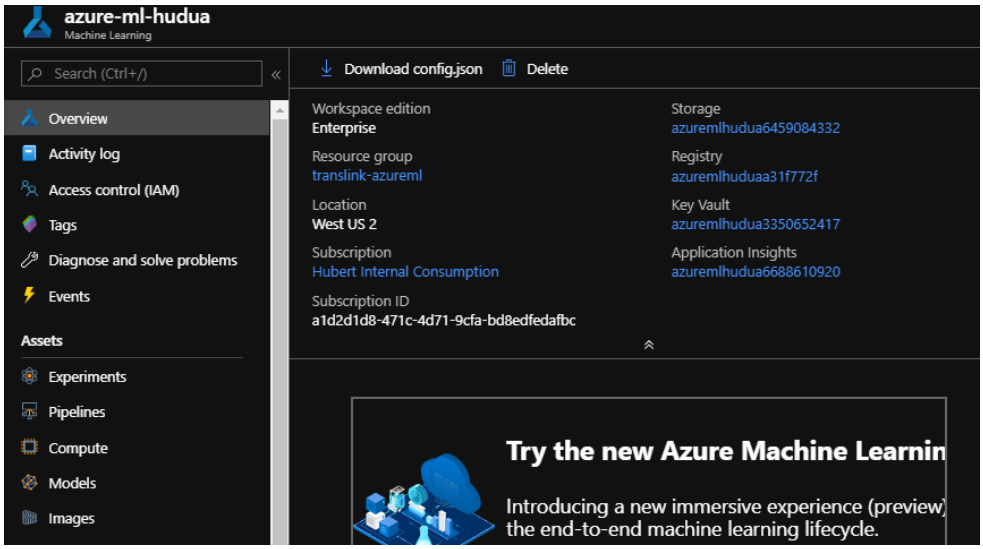
- Click on the Deploy to AKS task and verify the service connection is configured correctly

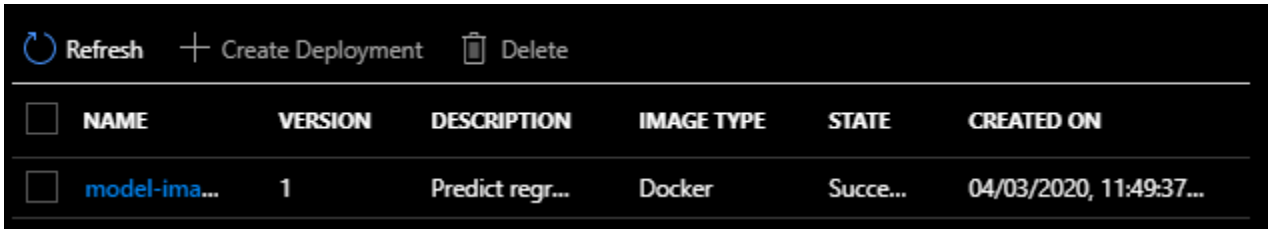


5. ML Ops

Azure DevOps integration

- Just before running the release pipeline, go to the Azure ML service in Azure Portal and verify the model image has been successful created
- It is under Assets → Images

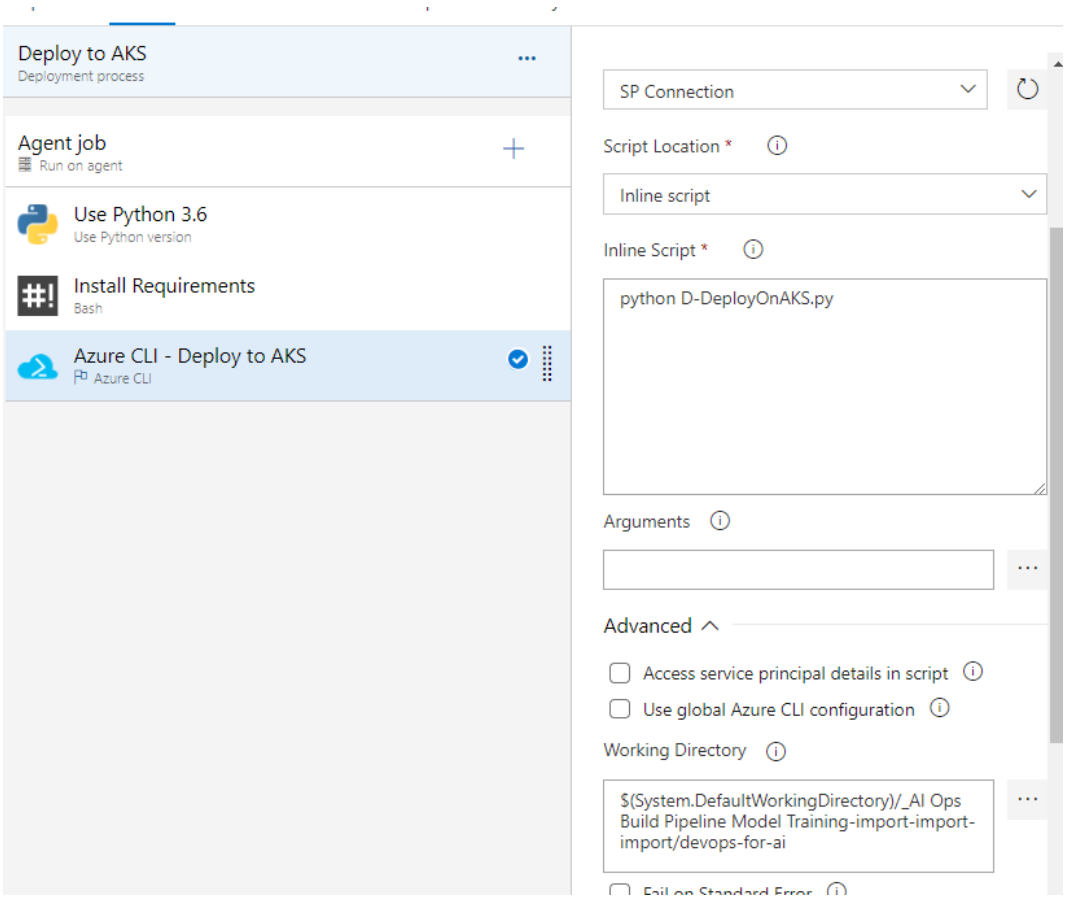




5. ML Ops

Azure DevOps integration

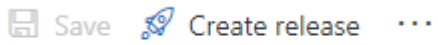
- Click on the Deploy to AKS task and verify the service connection is configured correctly
- And then the Advanced → Working Directory is selected to devops-for-ai correctly from the artifacts



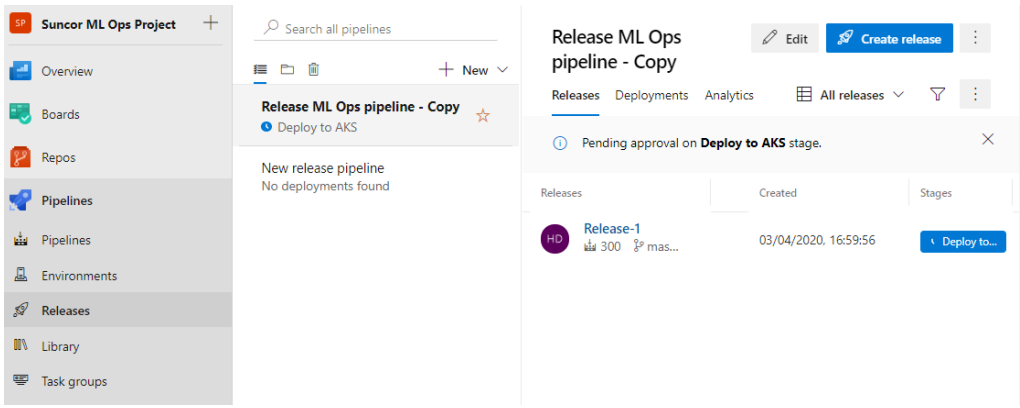
5. ML Ops

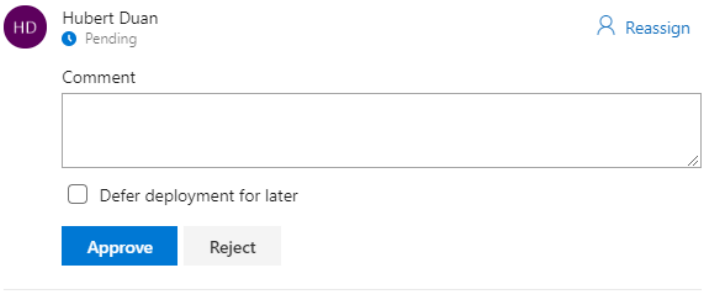
Azure DevOps integration

- Going back to Azure DevOps release pipelines, verify it is saved and click on create release



- Of course the approver should get an email so approve it. You can also approve through clicking Pipelines → Releases → Deploy



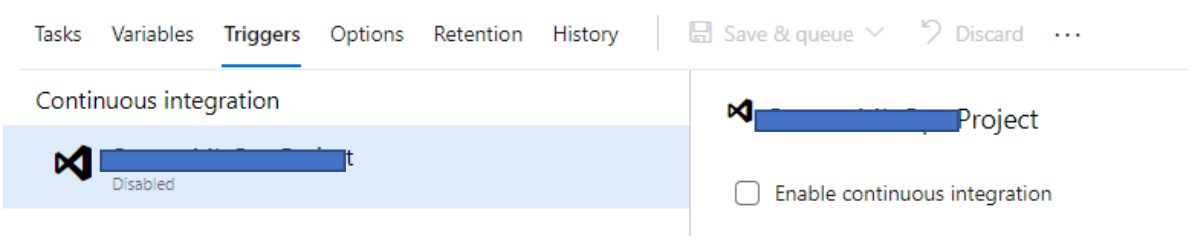


5. ML Ops

Azure DevOps integration

You can of course set up continuous integration and continuous deployment

- For continuous integration, go to Pipelines → Pipelines
- Select the pipeline and click on Edit
- Go to Triggers and click on Enable CI



5. ML Ops

Azure DevOps integration

- For continuous integration, go to Pipelines → Releases
- Select the pipeline and click on Edit
- Go to the continuous deployment trigger and enable it