Azure Machine Learning - (DL)

Mutaz Abu Ghazaleh | AI/ML Specialist

26/04/2022

UTS Microsoft

Agenda (DL)

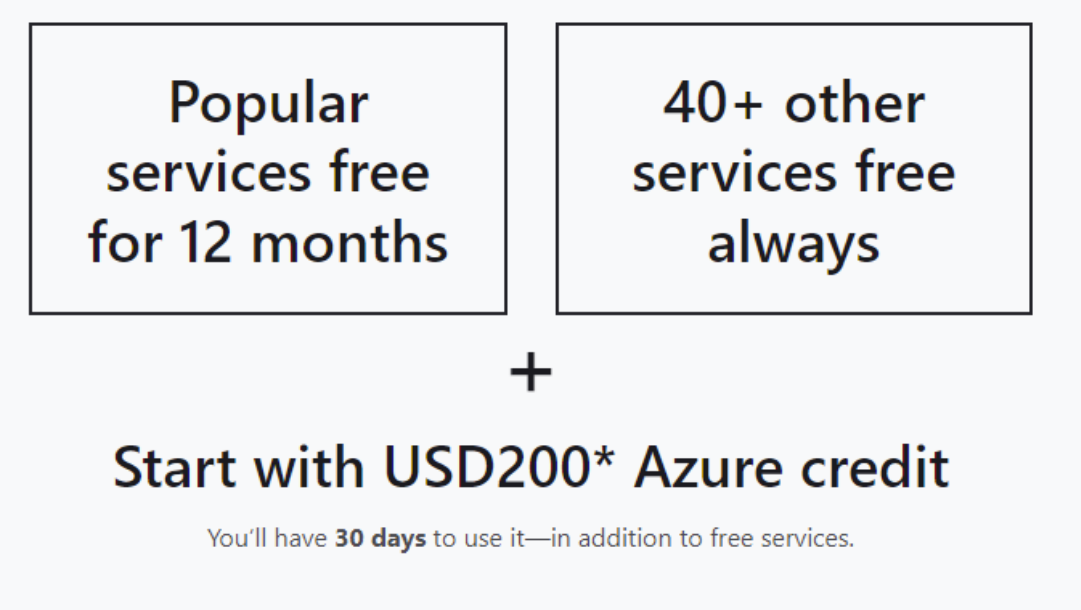
- Azure Portal
- Machine Learning Workspace
- Compute Instance – GPU
- Storage for saving images
- Notebooks and VS Code -> work with images
- Simple Cats/Dogs vision dataset

- Experiments and Tracking
- Saving model artifacts
- Jobs -> advanced (time permitting)

Get Azure

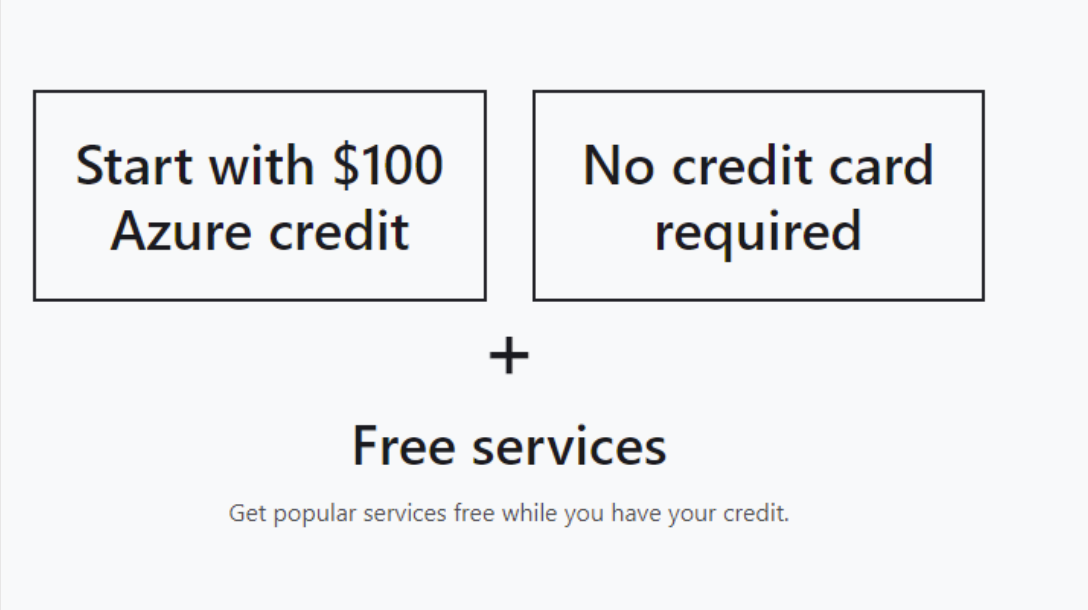
Get Azure

Azure Free Account



https://azure.microsoft.com/en-au/free/

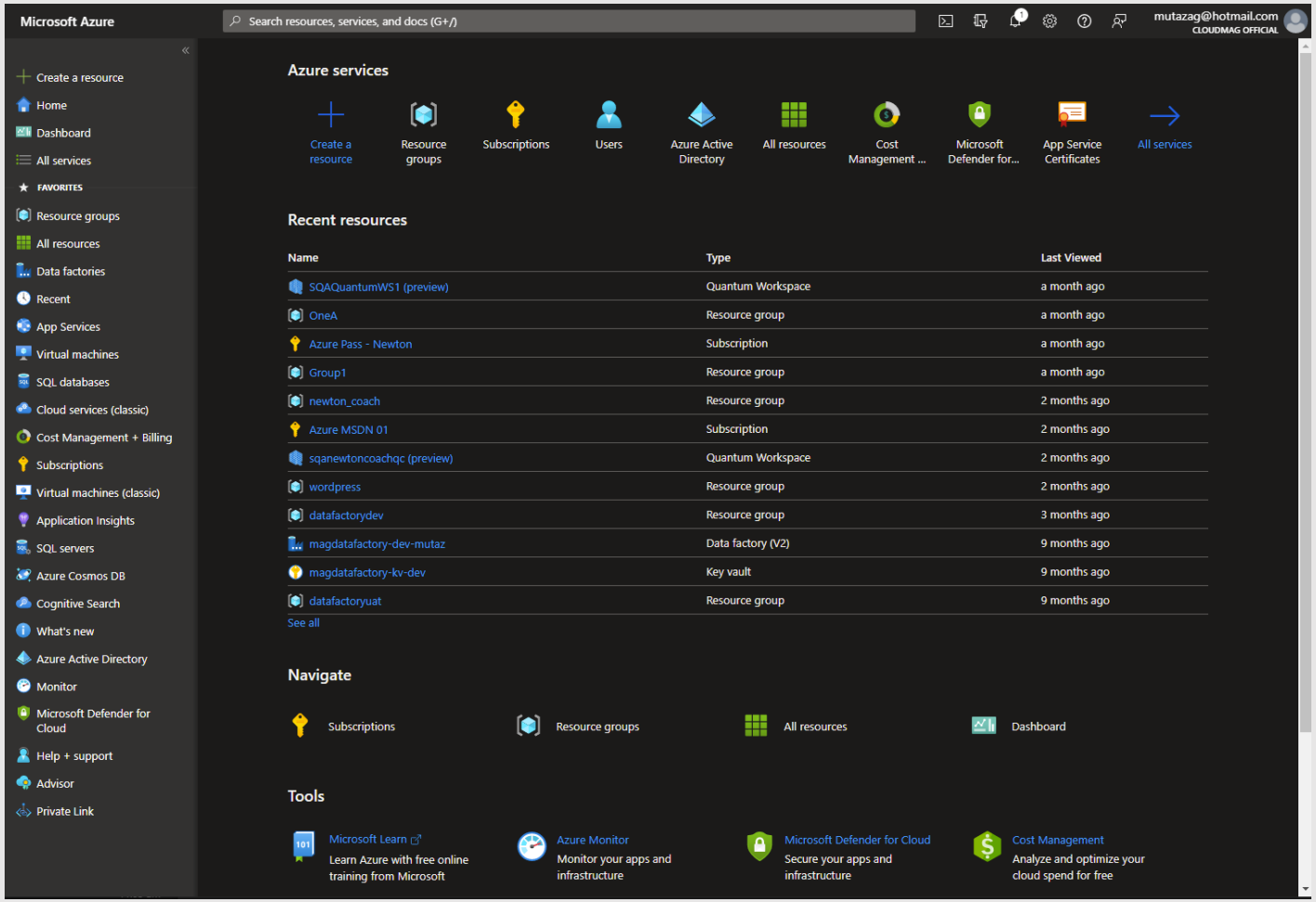
Azure for Students



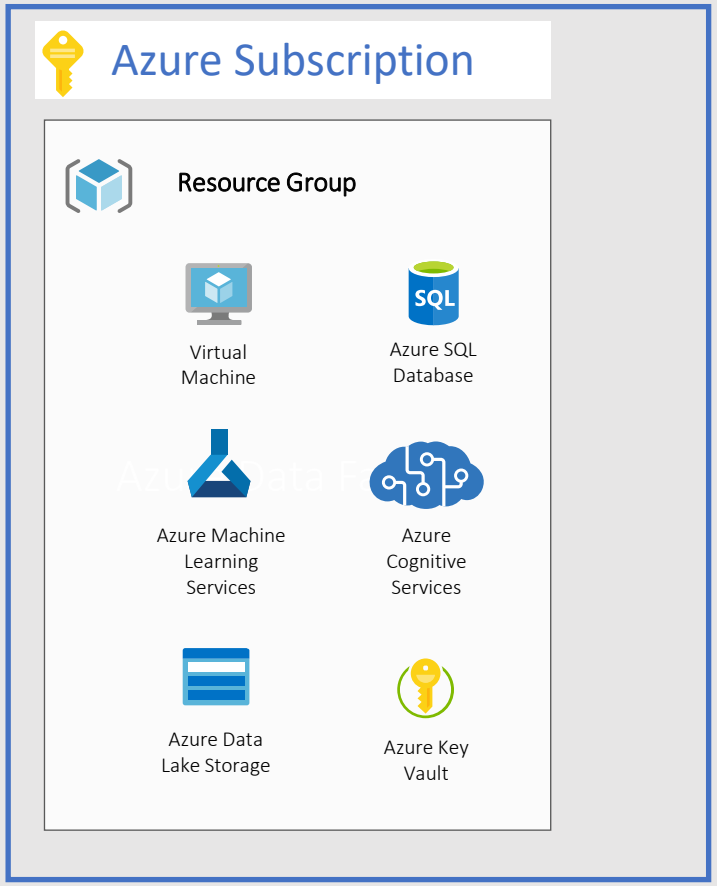
https://azure.microsoft.com/en-us/free/students/

Azure Portal

- A single portal to access all applications in your Azure Subscription
- Build, manage and monitor Azure resources
- Login with an Azure account: https://poral.azure.com



Hierarchy in Azure



Azure account: The email address that you provide when you create an Azure subscription is the Azure account for the subscription.

Subscription: A logical container for your resources. Each Azure resource is associated with only one subscription. Creating a subscription is the first step in adopting Azure.

Resource groups: Logical containers that you use to group related resources in a subscription. They're commonly used to represent a collection of assets that are required to support a workload, application, or specific function within a subscription.

Resources: An entity that's managed by Azure. Examples include Azure Virtual Machines, Machine Learning Services, SQL Database and storage accounts.

Azure Products

- AI + Machine Learning
- Analytics
- Compute
- Containers
- Databases
- Developer Tools
- DevOps
- Hybrid + multicloud
- Identity
- Integration
- Internet of Things
- Management and Governance
- Media
- Migration
- Mixed Reality
- Mobile
- Networking
- Security
- Storage
- Virtual desktop infrastructure
- Web

AI + Machine Learning

Anomaly Detector
Easily add anomaly detection capabilities to your apps

Azure Applied AI Services
Specialized services that enable organizations to accelerate time to value in applying AI to solve common scenarios

Azure Bot Service
Intelligent, serverless bot service that scales on demand

Azure Cognitive Search
AI-powered cloud search service for mobile and web app development

Azure Cognitive Services
Add smart API capabilities to enable contextual interactions

Azure Databricks
Fast, easy, and collaborative Apache Spark-based analytics platform

Azure Machine Learning
Bring AI to everyone with an end-to-end, scalable, trusted platform with experimentation and model management

Azure Open Datasets
Cloud platform to host and share curated open datasets to accelerate development of machine learning models

Azure Video Analyzer (preview)
Build intelligent video-based applications using the AI of your choice

Azure Video Analyzer for Media
Unlock video insights

Computer Vision
Distill actionable information from images

Content Moderator
Automated image, text, and video moderation

Custom Vision
Easily customize your own state-of-the-art computer vision models for your unique use case

Data Science Virtual Machines
Rich pre-configured environment for AI development

Face
Detect, identify, analyze, organize, and tag faces in photos

Form Recognizer
The AI-powered document extraction service that understands your forms

Health Bot
A managed service purpose-built for development of virtual healthcare assistants.

Immersive Reader
Empower users of all ages and abilities to read and comprehend text

Kinect DK
Build computer vision and speech models using a developer kit with advanced AI sensors

Language Understanding
Teach your apps to understand commands from your users

Metrics Advisor
An AI service that monitors metrics and diagnoses issues

Microsoft Genomics
Power genome sequencing & research insights

Personalizer
An AI service that delivers a personalized user experience

Project Bonsai (Preview)
A machine teaching service for creating intelligent industrial control systems using simulations

QnA Maker
Distill information into conversational, easy-to-navigate answers

Speaker Recognition (Preview)
Use speech to identify and verify individual speakers

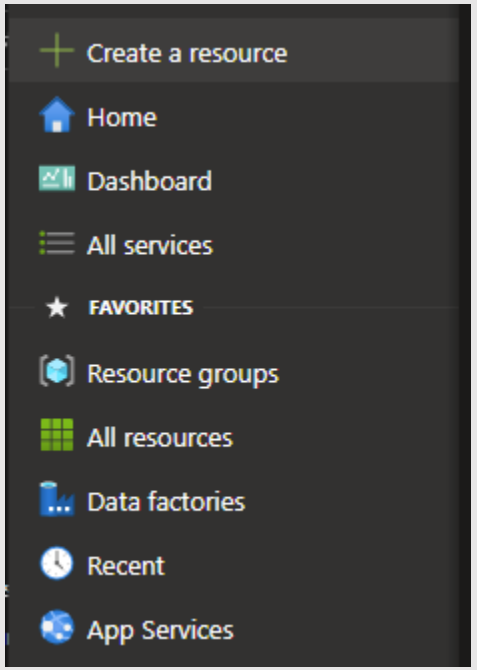
Speech to Text
Convert spoken audio to text for more natural interactions

Speech Translation
Easily integrate real-time speech translation to your app

Text Analytics
Easily evaluate sentiment and topics to understand what users want

Text to Speech
Convert text to speech to create more natural, accessible interfaces

Translator
Easily conduct machine translation with a simple REST API call



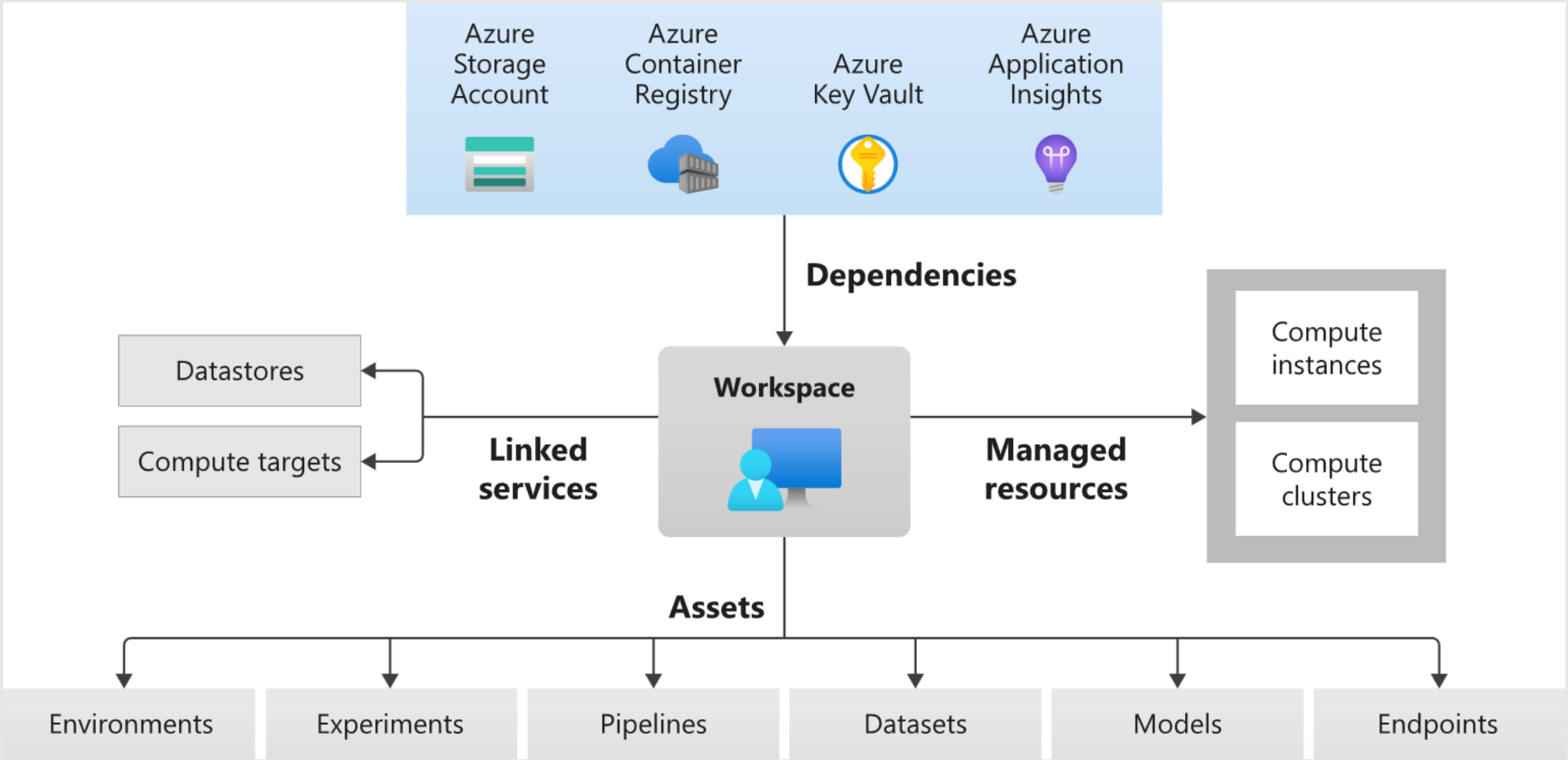
Exercise: Create Subscription and Workspace

- Create an Azure Subscription:
  - Claim your student subscription or get a free azure subscription

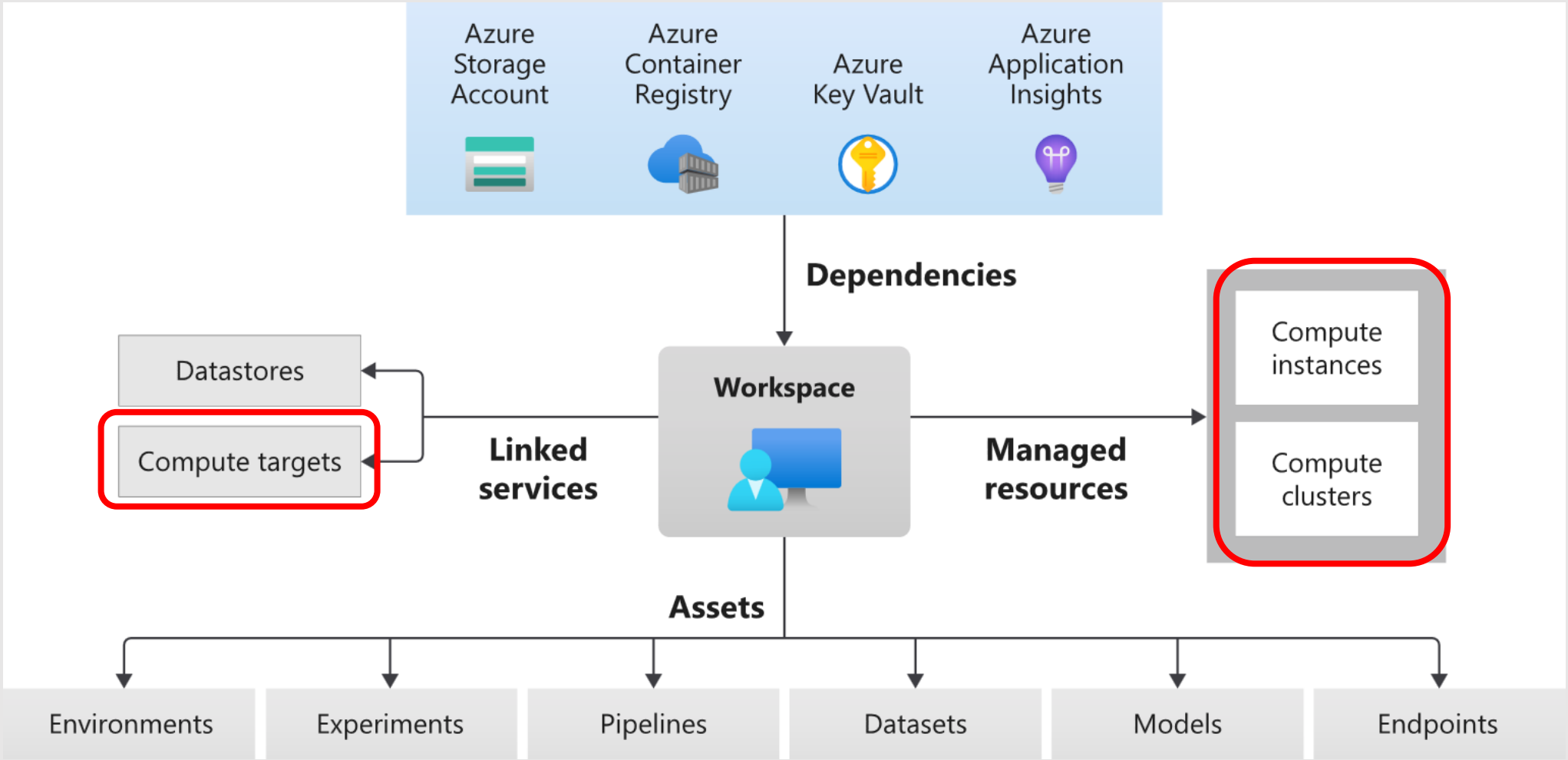
- Create a Machine Learning Workspace

Machine Learning Workspace

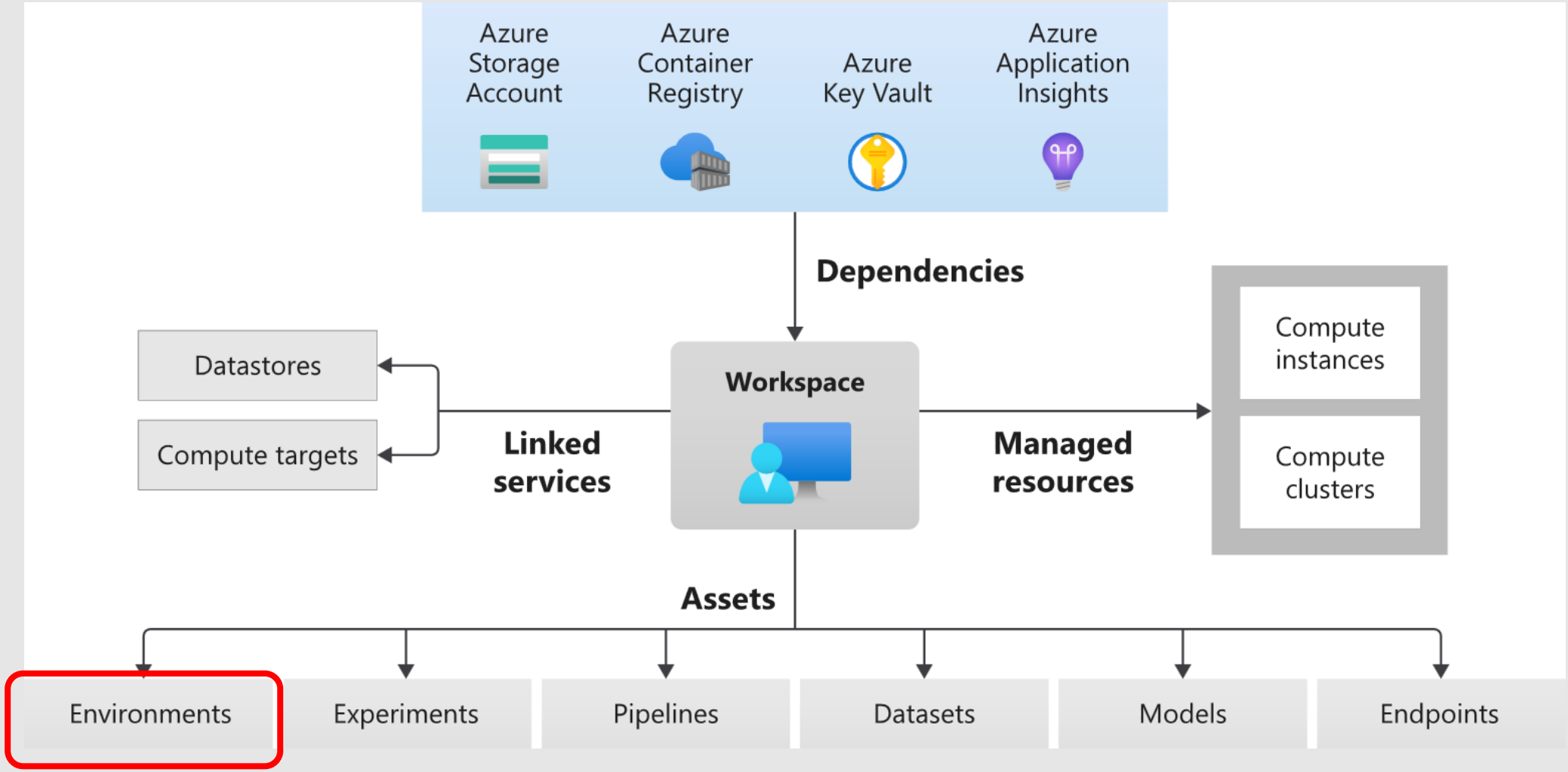
Machine Learning Workspace



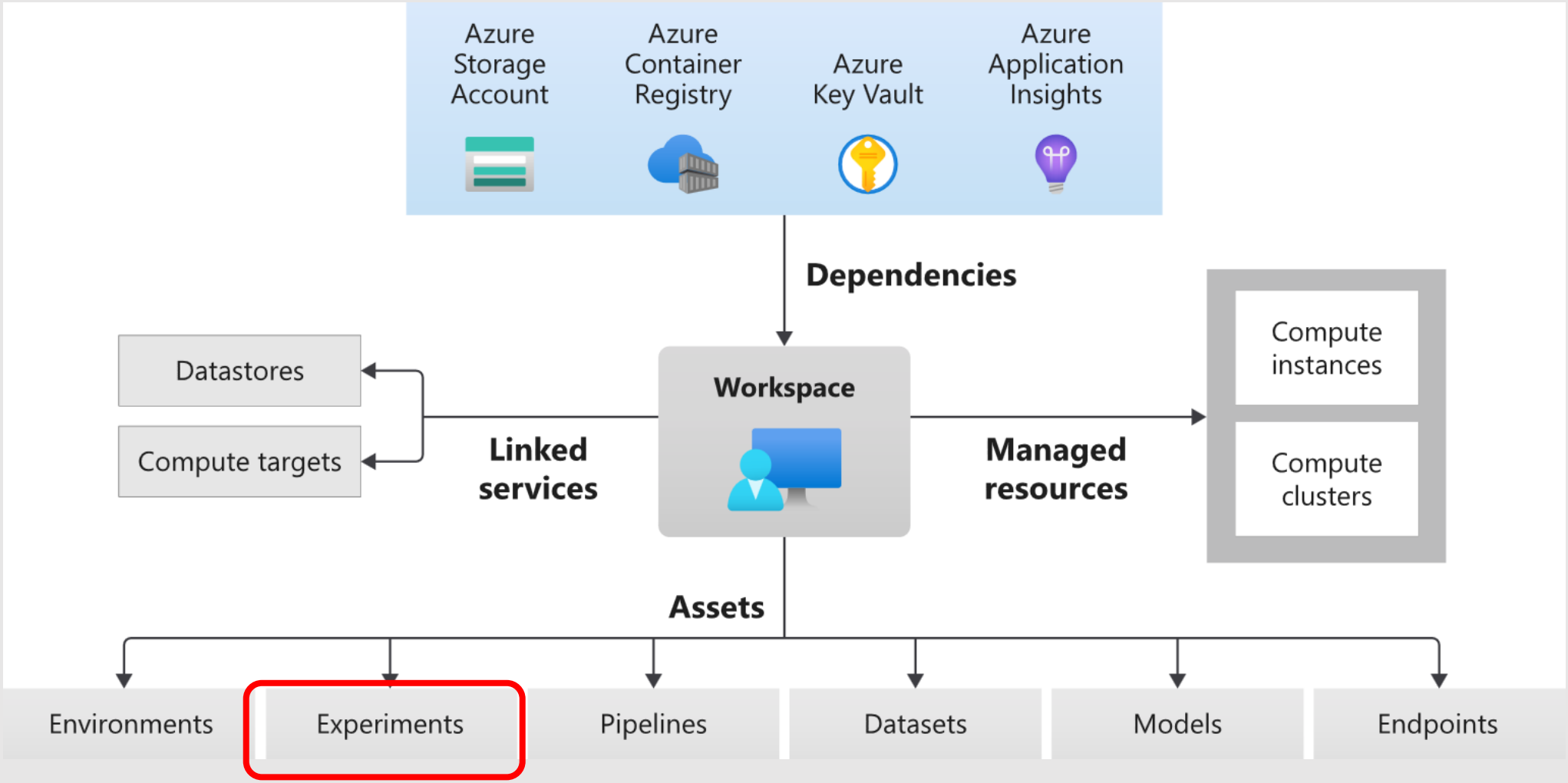
Compute


Azure Storage Account
Azure Container Registry
Azure Key Vault
Azure Application Insights
Dependencies
Datastores
Compute targets
Linked services
Workspace
Managed resources
Compute instances
Compute clusters
Assets
Environments
Experiments
Pipelines
Datasets
Models
Endpoints

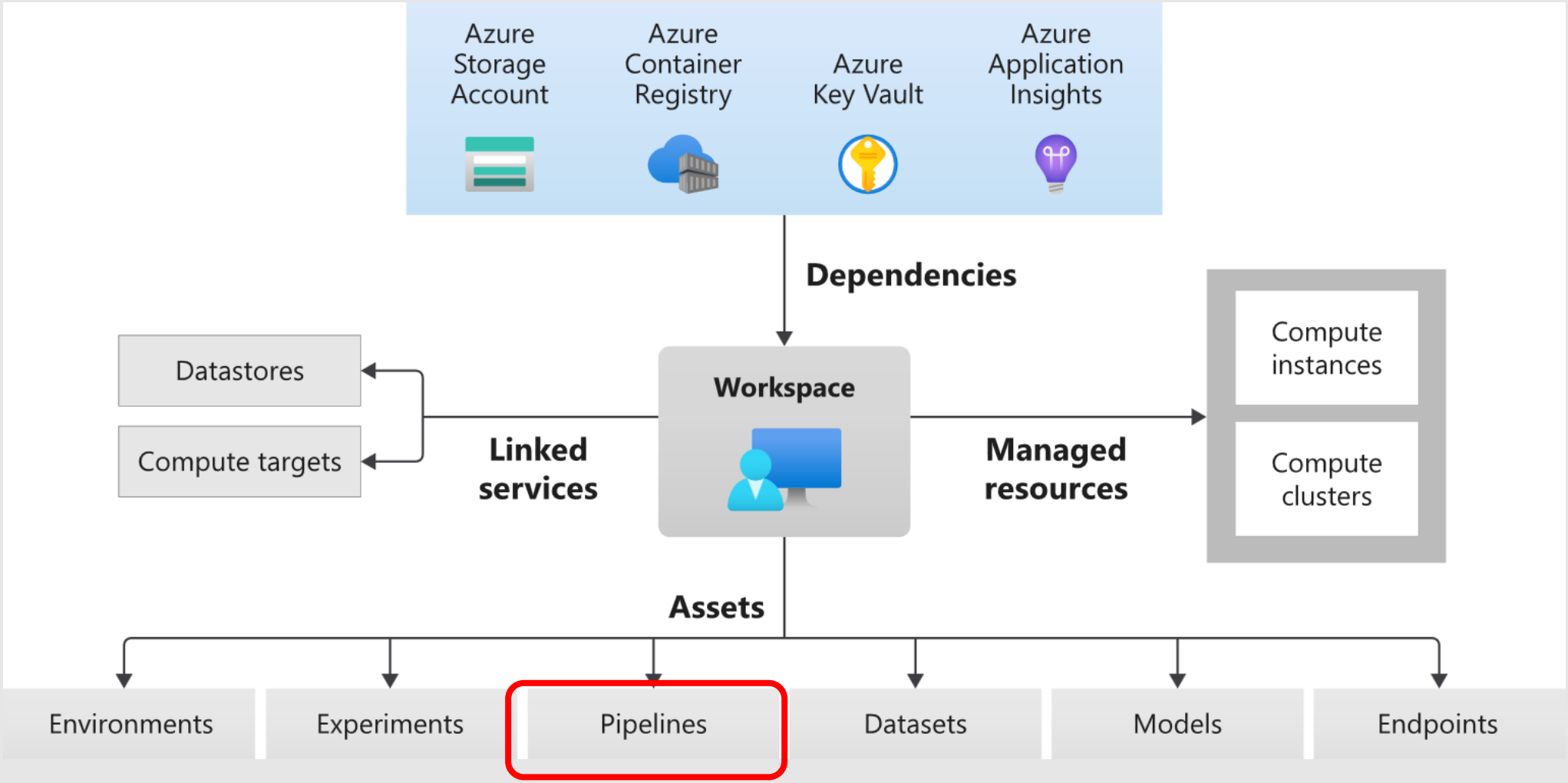
Environments



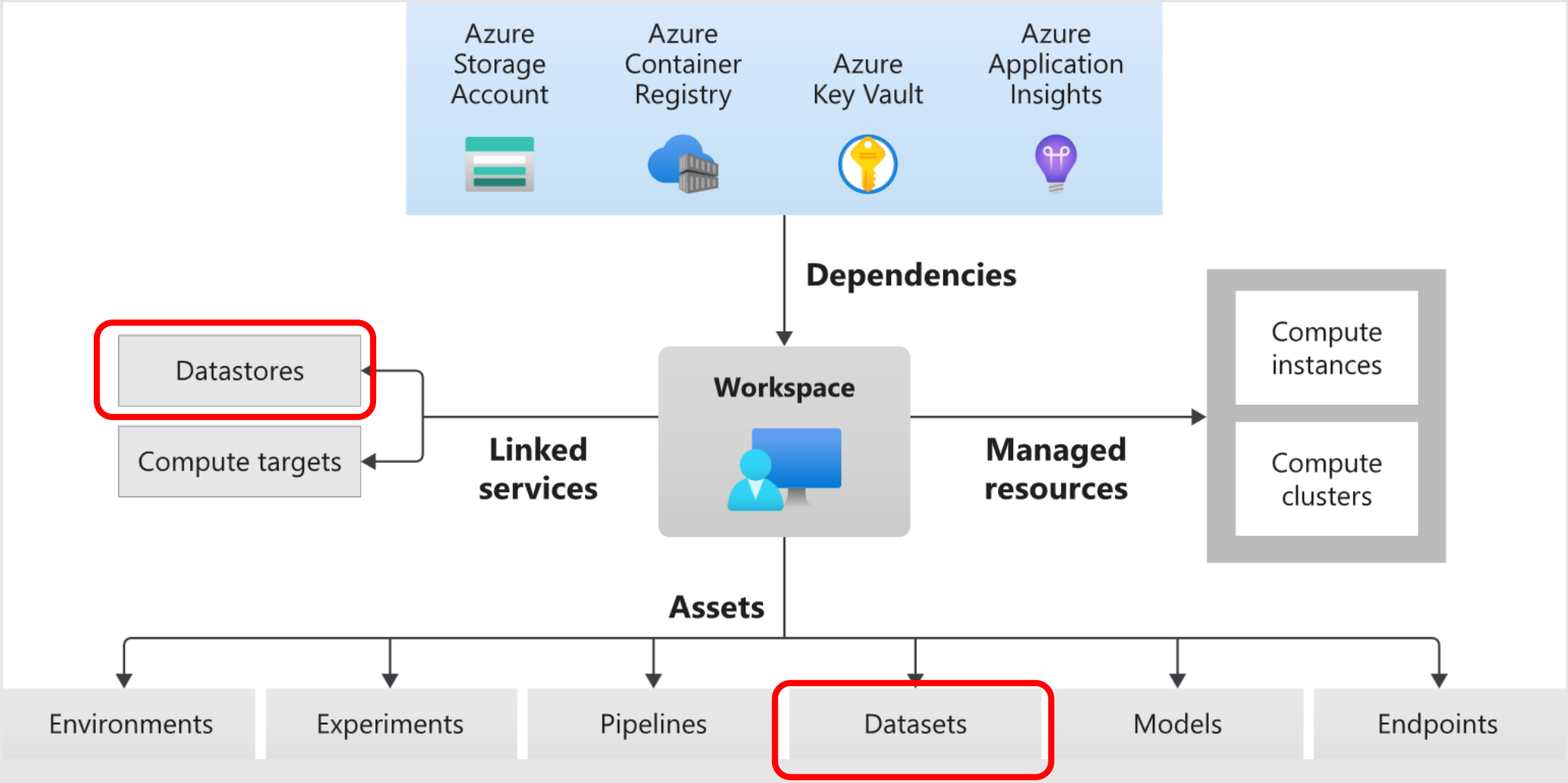
Experiments



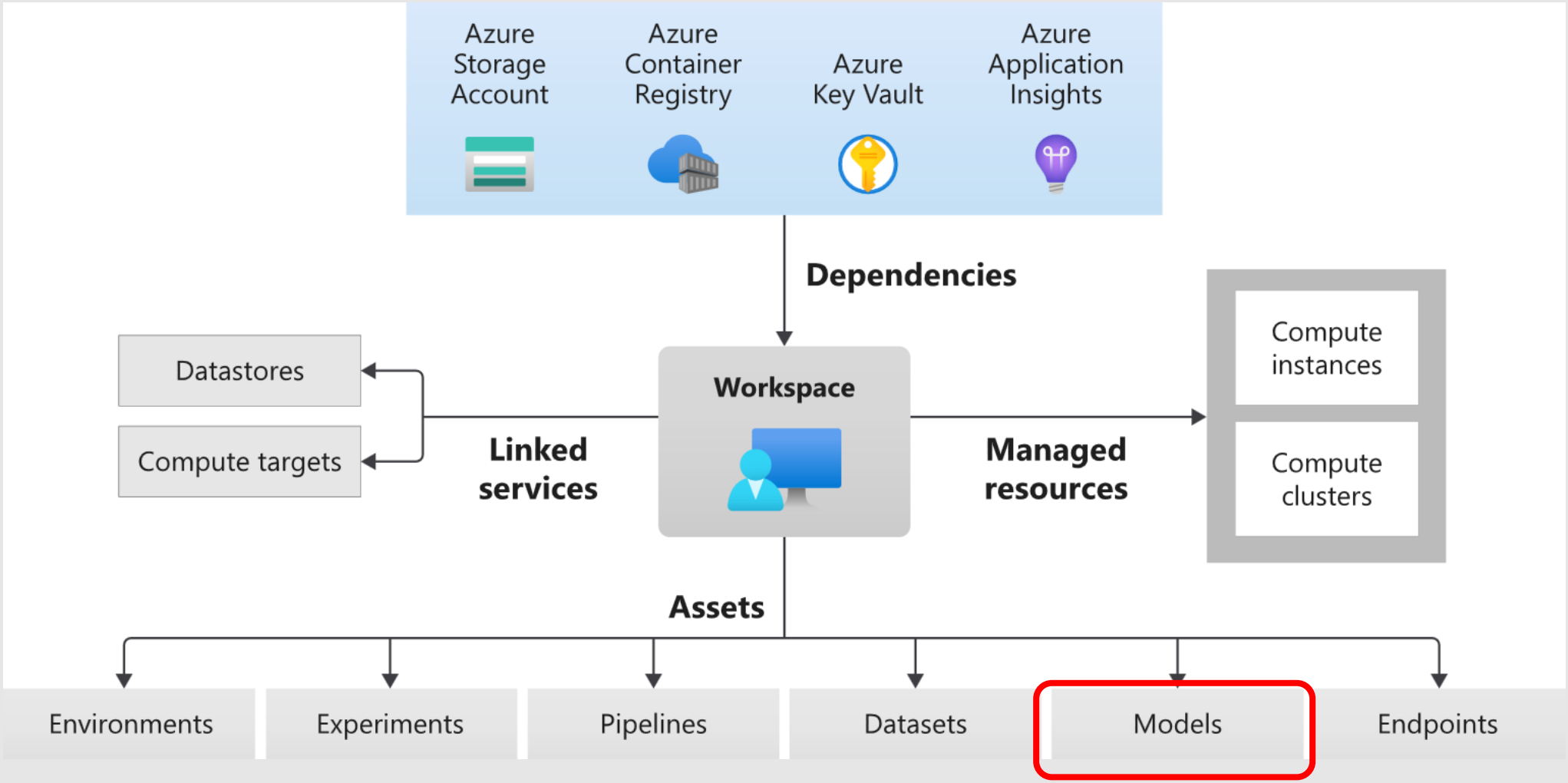
Pipelines


Azure Storage Account
Azure Container Registry
Azure Key Vault
Azure Application Insights
Dependencies
Datastores
Compute targets
Linked services
Workspace
Managed resources
Compute instances
Compute clusters
Assets
Environments
Experiments
Pipelines
Datasets
Models
Endpoints

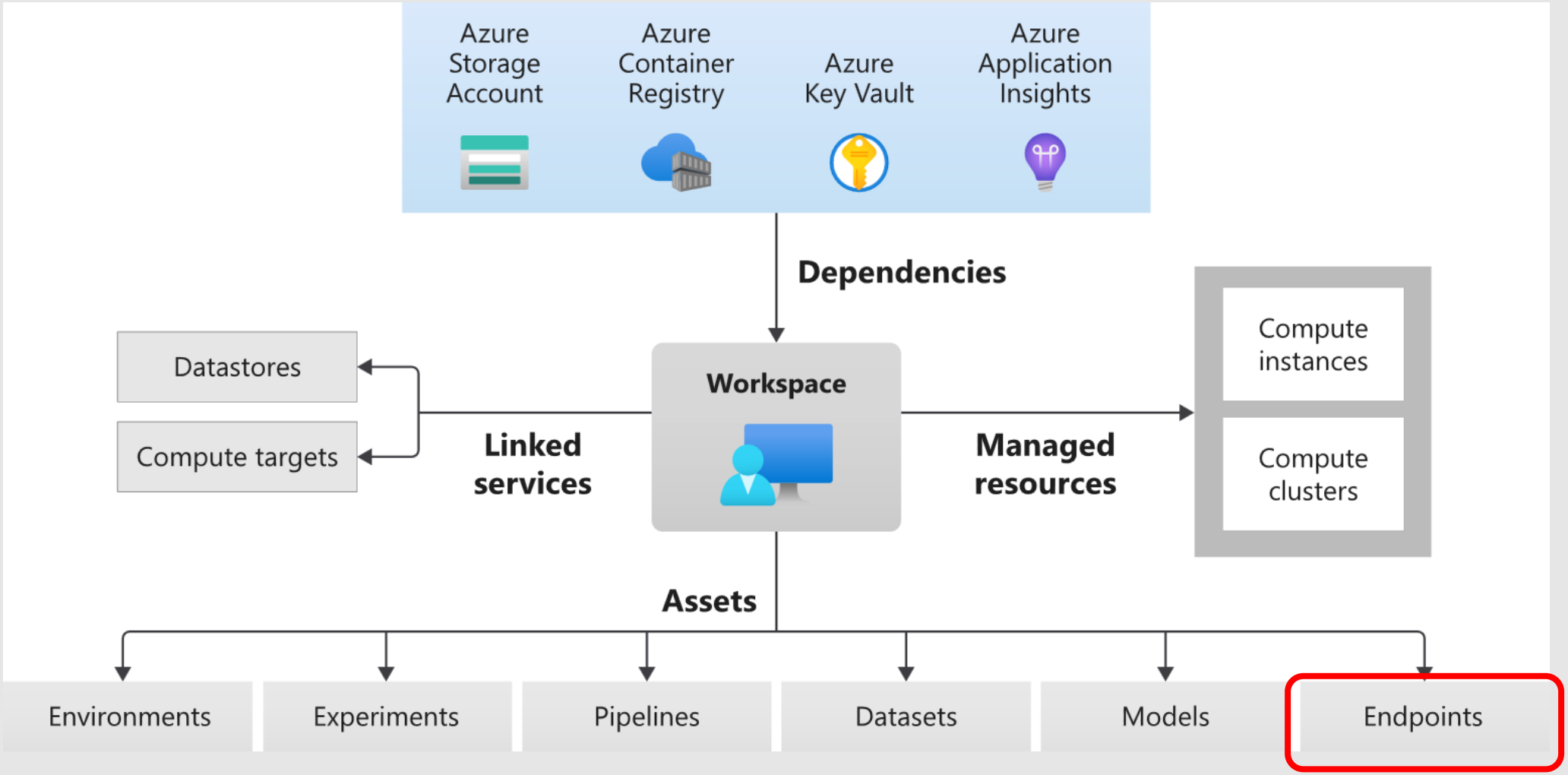
Datastores and Datasets



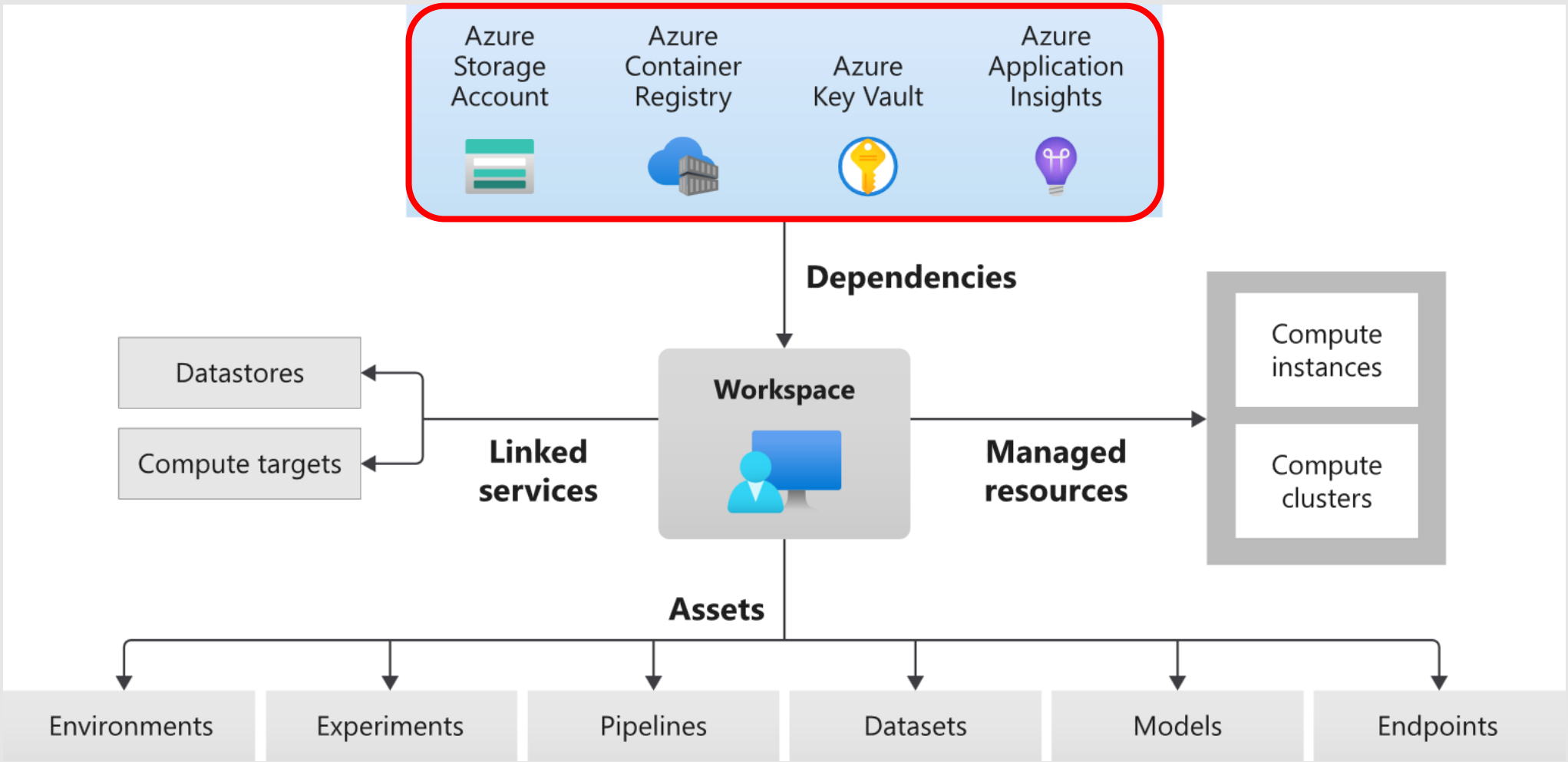
Models


Azure Storage Account
Azure Container Registry
Azure Key Vault
Azure Application Insights
Dependencies
Datastores
Compute targets
Linked services
Workspace
Managed resources
Compute instances
Compute clusters
Assets
Environments
Experiments
Pipelines
Datasets
Models
Endpoints

Endpoints


Azure Storage Account
Azure Container Registry
Azure Key Vault
Azure Application Insights
Dependencies
Datastores
Compute targets
Linked services
Workspace
Managed resources
Compute instances
Compute clusters
Assets
Environments
Experiments
Pipelines
Datasets
Models
Endpoints

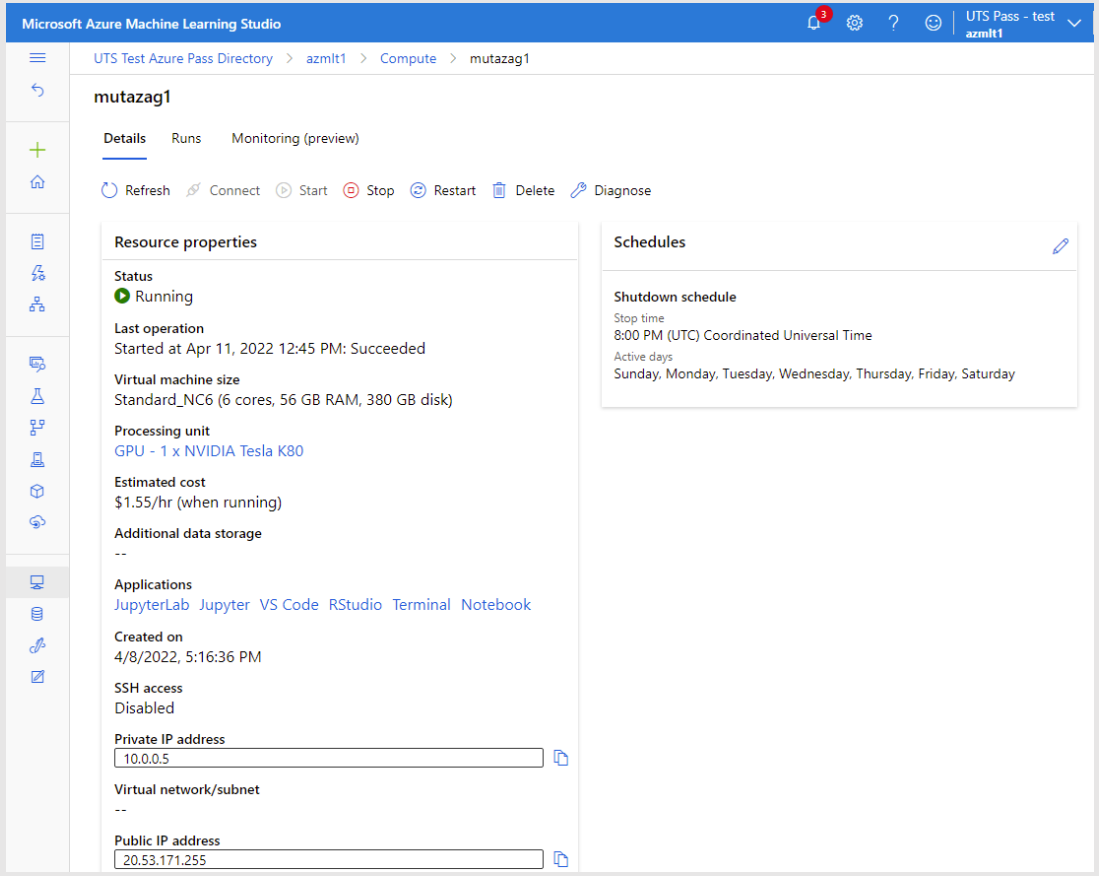
Workspace Services



Compute

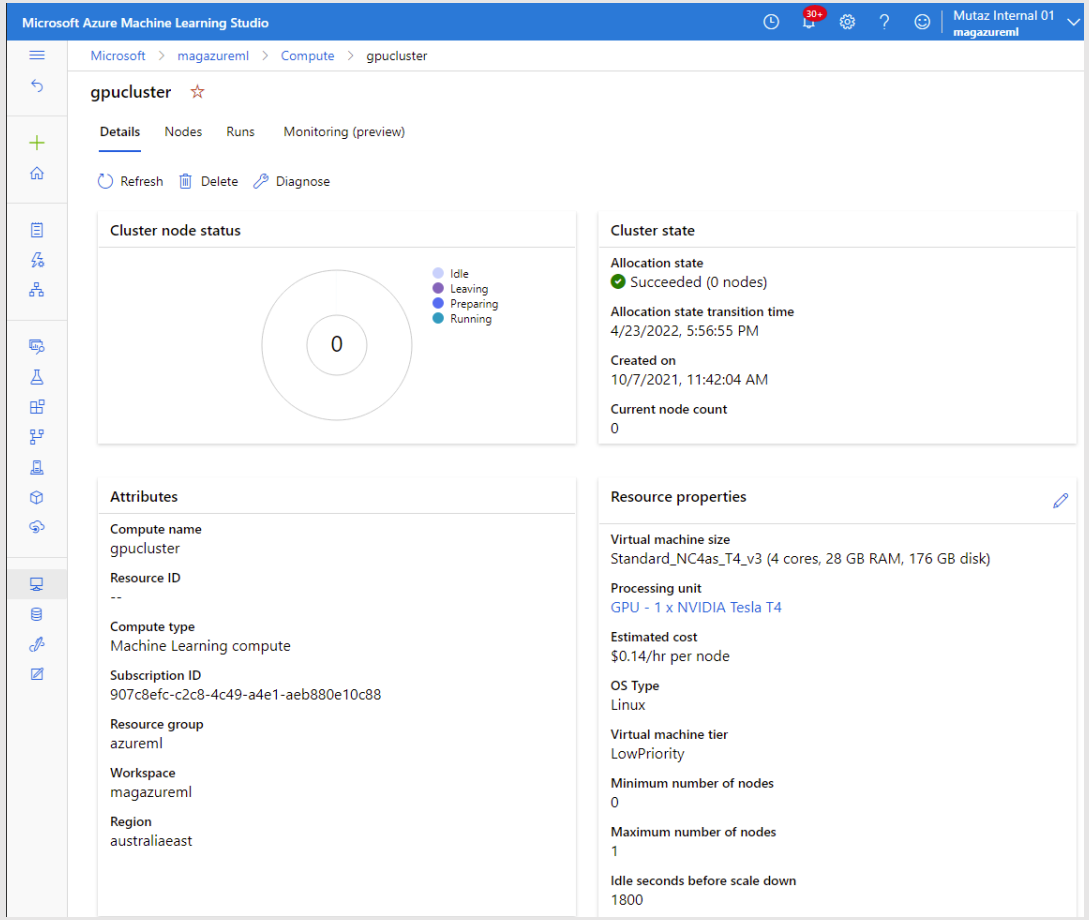
Compute Instance

- Provision a compute instance
  - STANDARD_DS3_V2 (4 cores, 14 GB RAM, 28 GB disk)
  - GPU: Standard_NC4as_T4_v3 or Standard_NC6
  - Set shutdown schedule
- Use for interactive development in:
  - VSCode
  - Jupyter
  - Notebook
- Submit long running jobs using Azure Machine Learning SDK or CLI

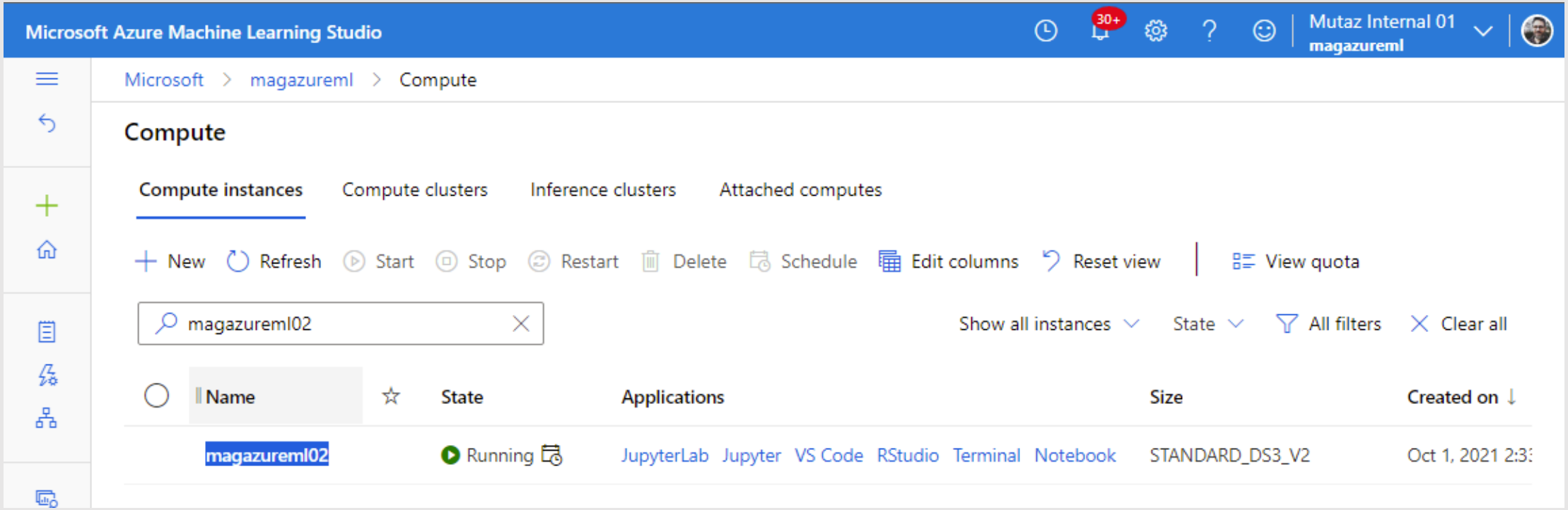


Compute Cluster

- Provision with GPU options: Standard_NC4as_T4_v3 or Standard_NC6
- Specify min and max number of nodes in a cluster
- Use for running jobs or training pipelines
- Submit long running jobs using Azure Machine Learning SDK or CLI



Applications on Compute Instances



Exercise: Create Compute Instance

- Create Compute Instance, and optionally a GPU cluster (1 node)
- Launch Terminal on CI and run:

	Command	Result
Conda environments	`conda env list`	Tensorflow env is azureml_py38_PT_TF
Azure cli version	`az version -o table`	We want version >= 2.15.0
Azure ML cli version	`az extension list -o table`	Looking for cli v2
Git version	`git version`	
Configured defaults	`az configure`	[defaults] workspace = <workspace name> group = <resource group name>
Login to Azure	`az login --identity`	Cli context logged in to Azure
Get mlflow tracking URI	`az ml workspace show --query mlflow_tracking_uri`	"azureml://australiaeast.api.azureml.ms/mlflow/v1.0 …"
Show Storage Account Key	`az ml workspace list-keys --query userStorageKey`	"ePf …"

Exercise: Get Started with a Git Repo

- Launch Terminal
- Clone a git repo – suggest to clone into ~/cloudfiles/code
- Explore project folder structure

```
cd ~/cloudfiles/code
git clone https://github.com/mutazag/catsanddogs
```

Code Repo: https://github.com/mutazag/catsanddogs

Datasets

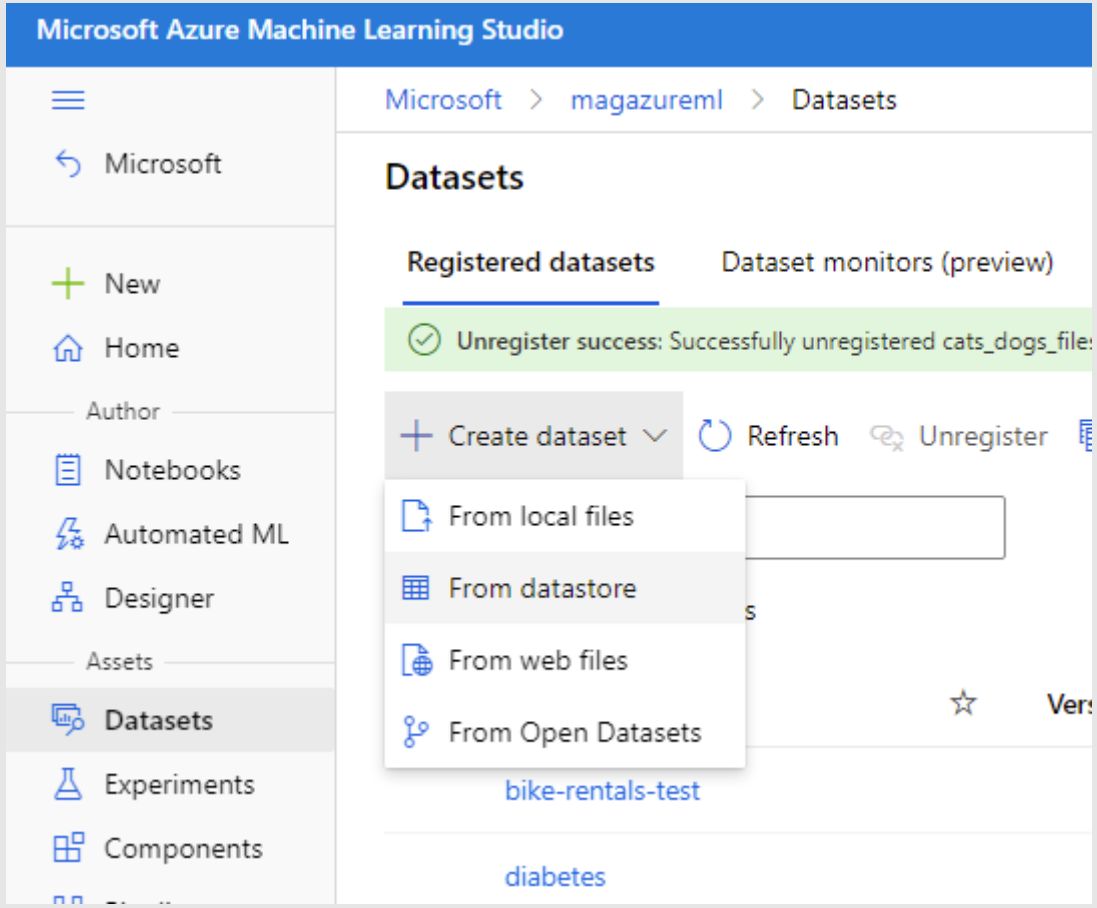
Exercise: Download Dataset

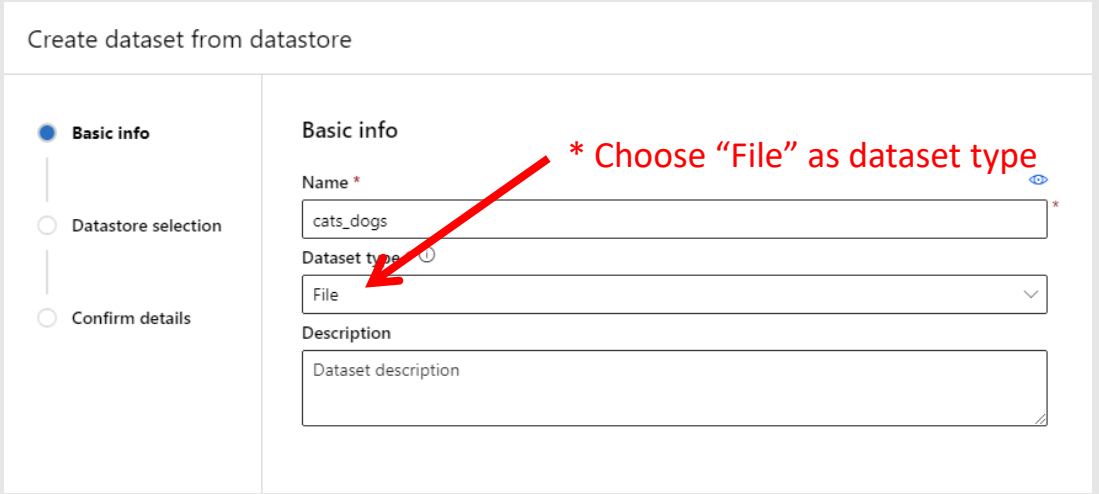
- Download the ‘Cats and Dogs’ dataset to compute instance
  - https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip
- Unzip the file
- Upload to Azure Storage Account
  - Use default storage account for the workspace
  - Create a storage container in storage account
  - Automate upload using cli for storage (storage-preview)
    - Requires: az extension add --name storage-preview
- Create and Register and Azure ML Dataset
  - Manually create the dataset
- Solution file: 01_download_files.sh:

Script – set variables to your storage account

```
# script requires azure cli extension for storage (storage-preview)
# az extension add --name storage-preview

# set variables:
resource_group=sina
account_name=godzillasinastorage
container_name=datasets

# constants:
zip_filename=cats_and_dogs_filtered.zip
local_download_dir=~/cloudfiles/data/
data_folder=cats_and_dogs_filtered
full_directory_name=$local_download_dir$data_folder
echo $full_directory_name

# download and unzip"
wget https://storage.googleapis.com/mledu-datasets/cats_and_dogs_filtered.zip  -P $local_download_dir

unzip  $local_download_dir$zip_filename -d $local_download_dir

chmod 777 $full_directory_name

# upload to azure storage
account_key=$(az storage account keys list -g $resource_group -n $account_name -o tsv --query "[0].{Value:value}")

az storage container create --name $container_name --auth-mode key --account-key $account_key --account-name $account_name -g $resource_group

az storage blob directory upload -c $container_name --auth-mode key --account-key $account_key --account-name $account_name -s $full_directory_name -d . --
recursive
```

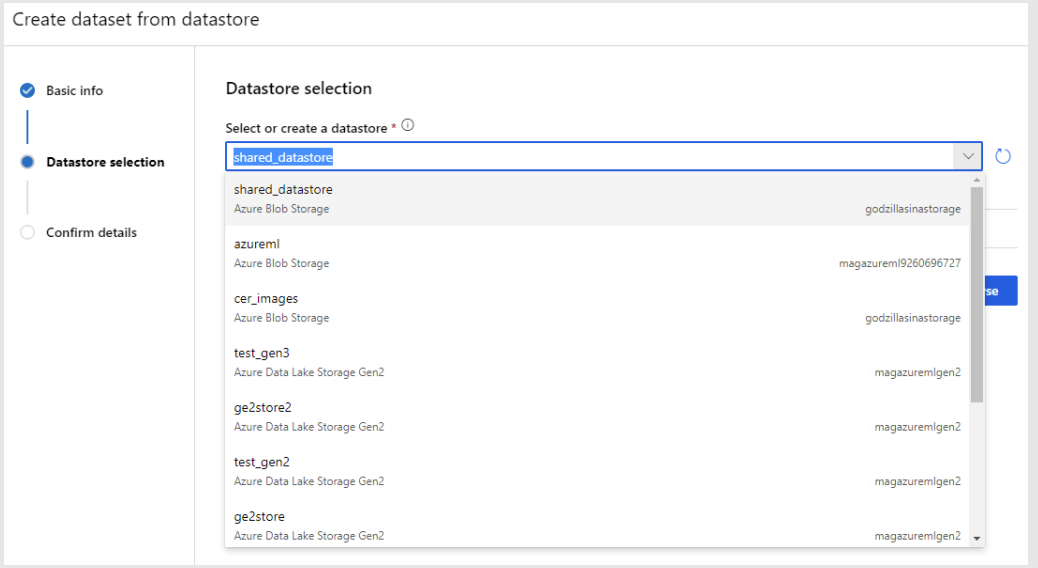
Azure ML Dataset

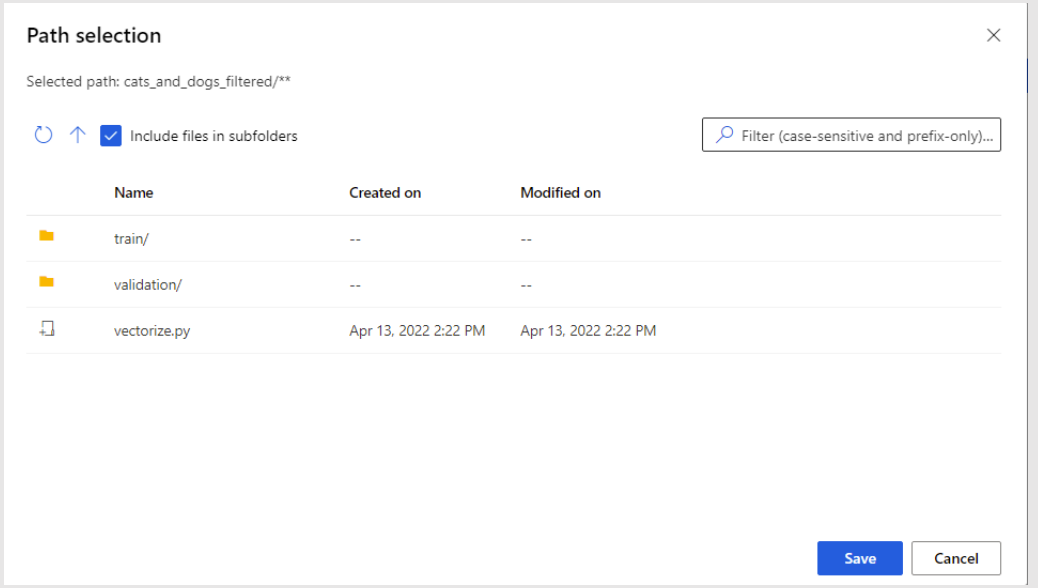




Create Dataset

- Select the data store
  - This is the storage account where you previously uploaded the downloaded cats and dogs data set
  - Create a data store in Azure ML if you don't have one yet
- Select the path of the data set in the data store
  - Ensure that path points to the location of the uploaded files
  - Select to include files in subfolders





Exercise: Consume Dataset

File Datasets

- Use Dataset object to mount and start a mount point on a compute instance

```
ds_mount = ds.mount(mount_point)
ds_mount.start()
```

- This option presents the dataset as a mounted drive in Linux

- Solution file: 02_explore_dataset.ipynb

Tabular Datasets

- Tabular datasets are created from structure data files, e.g.: csv or parquet

- A workspace dataset can be consumed as a pandas dataframe in code

```
df = ds.to_pandas_dataframe()
df.head()
```

- Other options include spark or dask dataframes

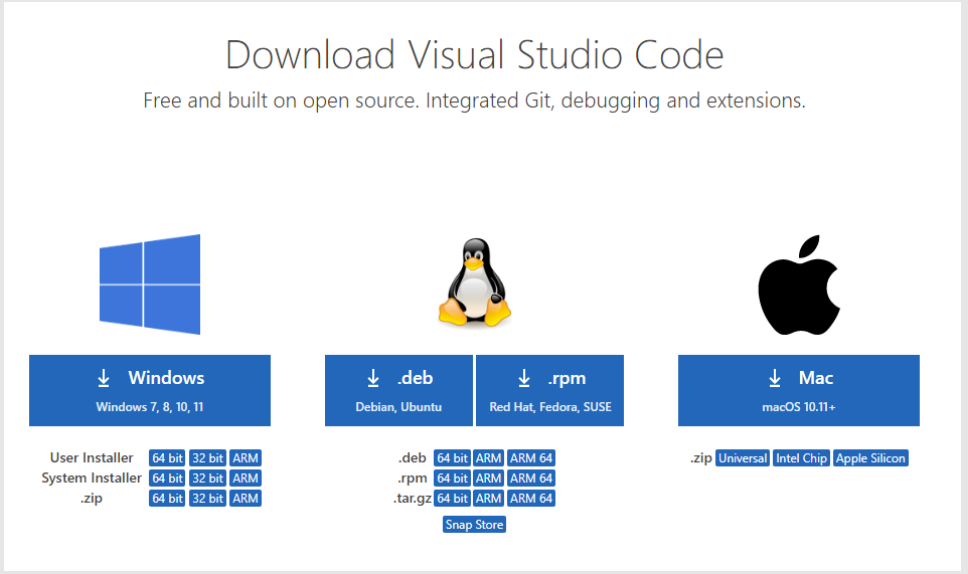
Development Environment

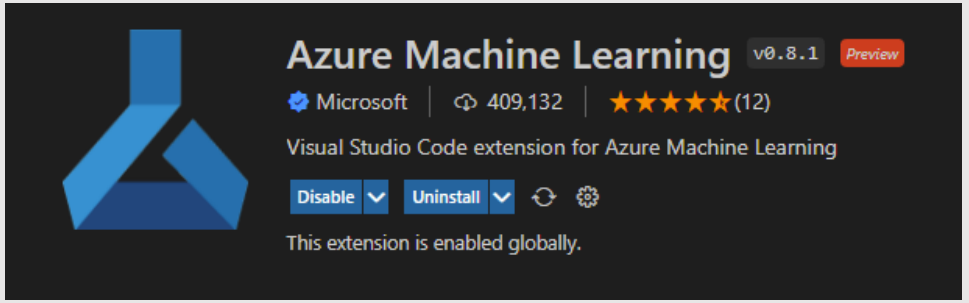
Development Environment Options

Environment	Pros	Cons
Local Environment	- Full control of your development environment and dependencies - Run with any build tool, environment or IDE of your choice	- Takes longer to setup and get started - Require installing SDKs and tools - Compute and storage limits
Remote on Azure ML Compute Instance	- Easy to get started, compute instance preconfigured with tools and libraries - AML SDK notebooks and tutorials preloaded - Scale compute and storage	- Manage cost for compute instance - More complex development environment setup

Development with VS Code

- Use VS Code and Azure ML Extension
- Write and debug code locally or execute remotely on a compute instance
- Run Jupyter notebooks from within VS Code against a local or remote Jupyter server
- Access AML workspace artefacts





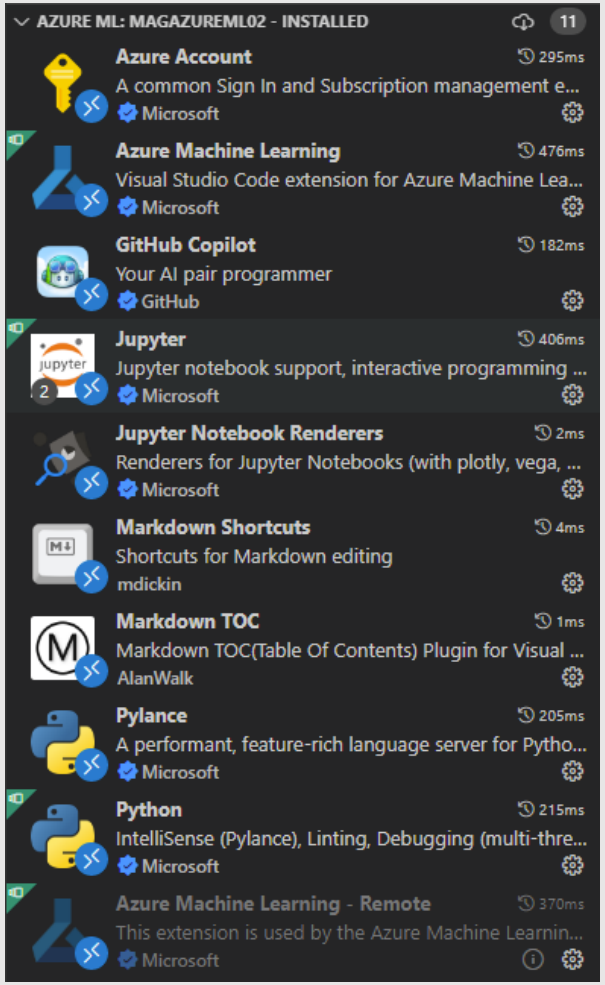
Install VS Code

Access to AML workspace and compute instance

- Install VS Code: Download Visual Studio Code - Mac, Linux, Windows
- Python: Download Python | Python.org

VS Code Extensions:

- Python
- Azure Machine Learning
- Azure Account
- Jupyter



Azure Machine Learning in Visua

https://code.visualstudio.com/docs/datascience/azure-machine-learning

Visual Studio Code | Docs | Updates | Blog | API | Extensions | FAQ | Learn | Search Docs | Download

Version 1.66 is now available! Read about the new features and fixes from March.

Overview

SETUP

GET STARTED

USER GUIDE

LANGUAGES

NODE.JS / JAVASCRIPT

TYPESCRIPT

PYTHON

JAVA

C++

CONTAINERS

DATA SCIENCE

- Overview
- Jupyter Notebooks
- Data Science Tutorial
- Python Interactive
- PyTorch Support
- Azure Machine Learning

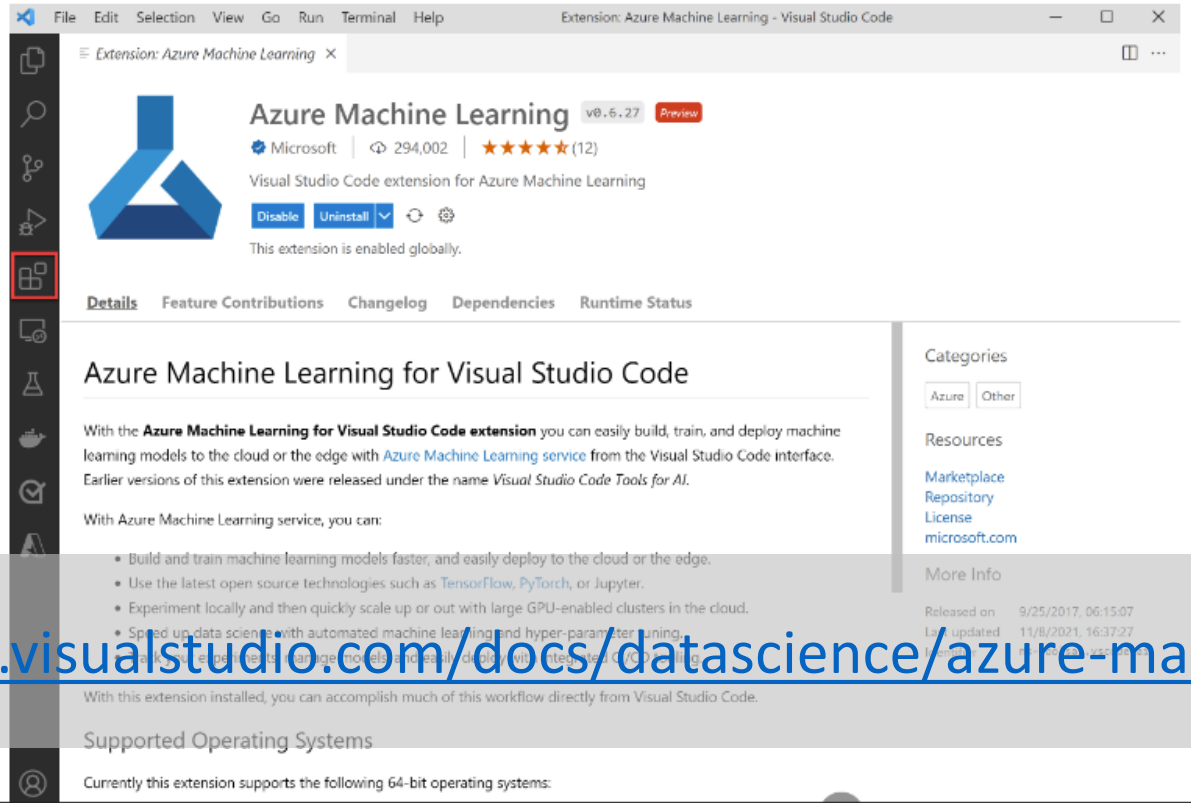
REMOTE

Azure Machine Learning in VS Code

Edit

Azure Machine Learning is a cloud-based environment you can use to train, deploy, automate, manage, and track machine learning models. For more information on Azure Machine Learning, see What is Azure Machine Learning?

The Azure Machine Learning VS Code extension lets you use the features you're used to in Visual Studio Code for developing your machine learning applications.



IN THIS ARTICLE

- Connect to remote compute instances
- Azure Machine Learning 2.0 CLI support (preview)
- Train machine learning models
- Manage resources
- Remote Jupyter servers
- Git integration
- Next steps

- Tweet this link
- Subscribe
- Ask questions
- Follow @code
- Request features
- Report issues
- Watch videos

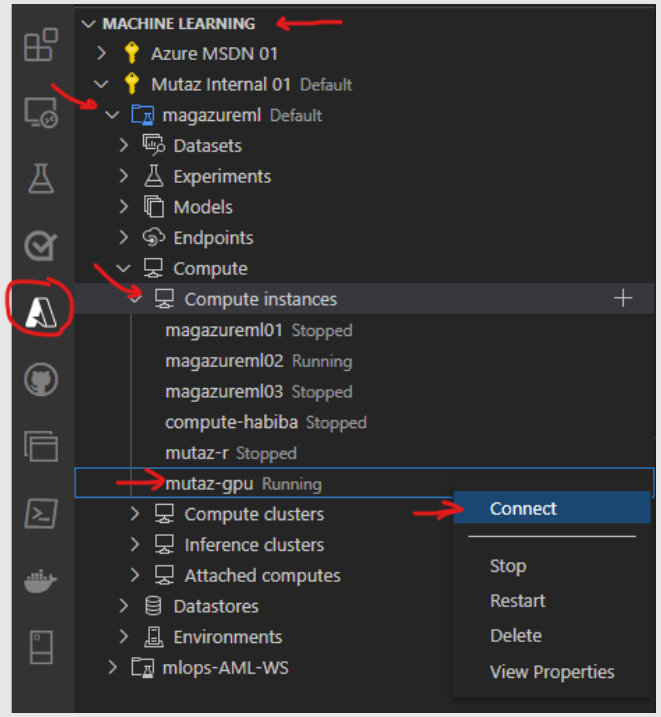
https://code.visualstudio.com/docs/datascience/azure-machine-learning

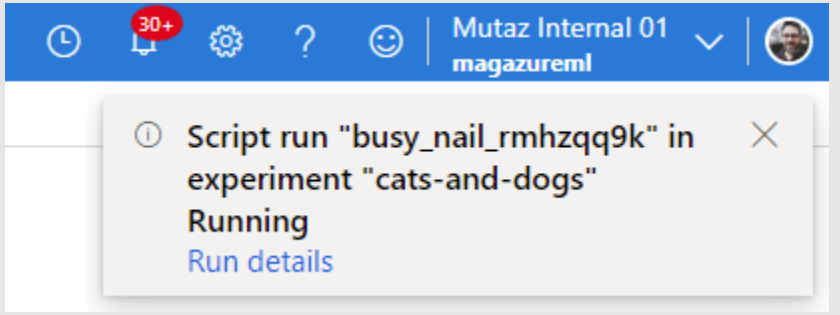
Get Started with Compute Instance

- Connect VS Code to compute instance
- Start a new terminal window in VS Code
- Open project folder (~/cloudfiles/code/catsanddogs)

- What is the name of your training experiment run in Azure ML Workspace?

- Run model training notebook: 03_train_model.ipynb





Experiments and Tracking

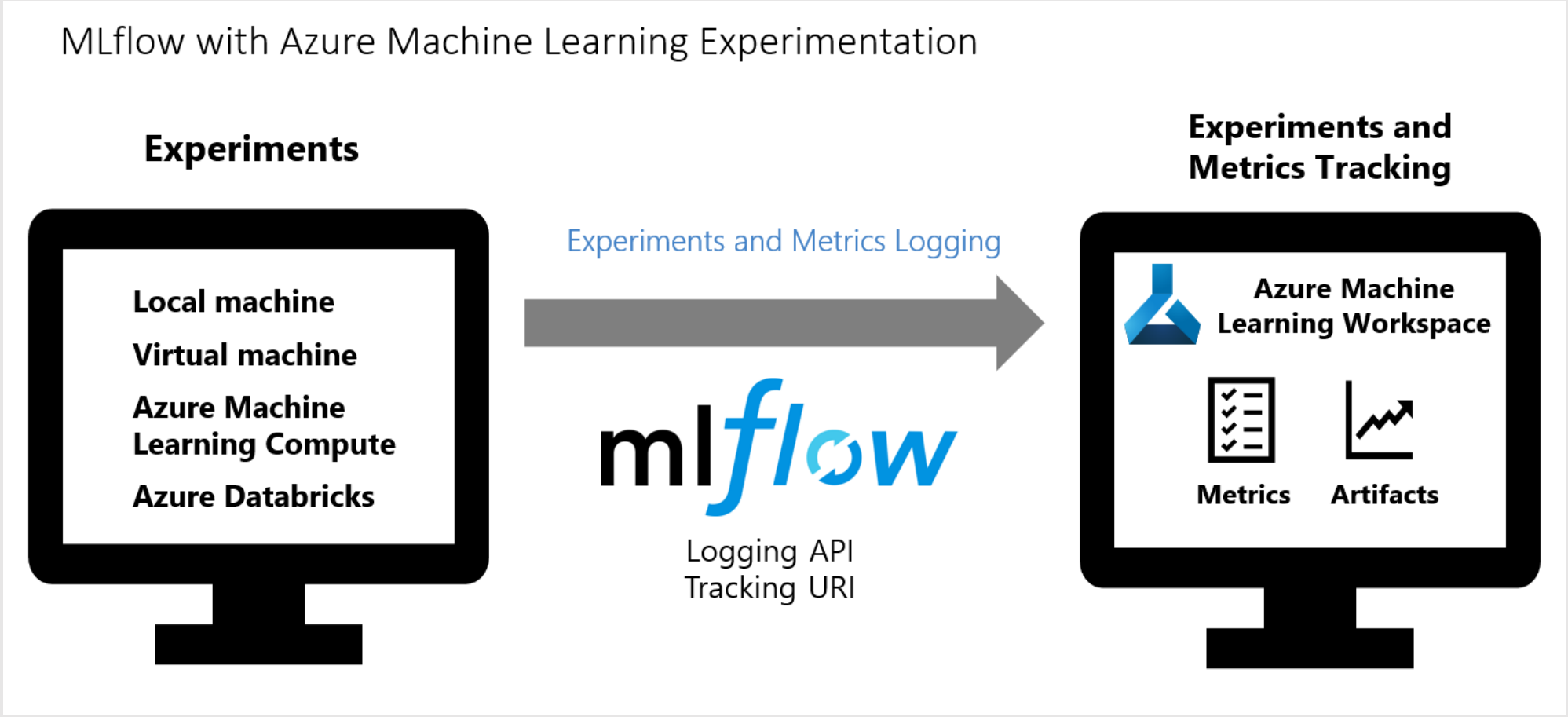
MLFlow for Tracking Experiments

MLflow is an open-source platform for managing the end-to-end machine learning lifecycle.

Mlflow tackles key functions of ML workloads. MLFlow can be used to track experiments to record and compare parameters and results.

MLflow is library-agnostic. You can use it with any machine learning library, and in any programming language, since all functions are accessible through a REST API and CLI. For convenience, the project also includes a Python API, R API, and Java API.

MLFlow and Azure ML

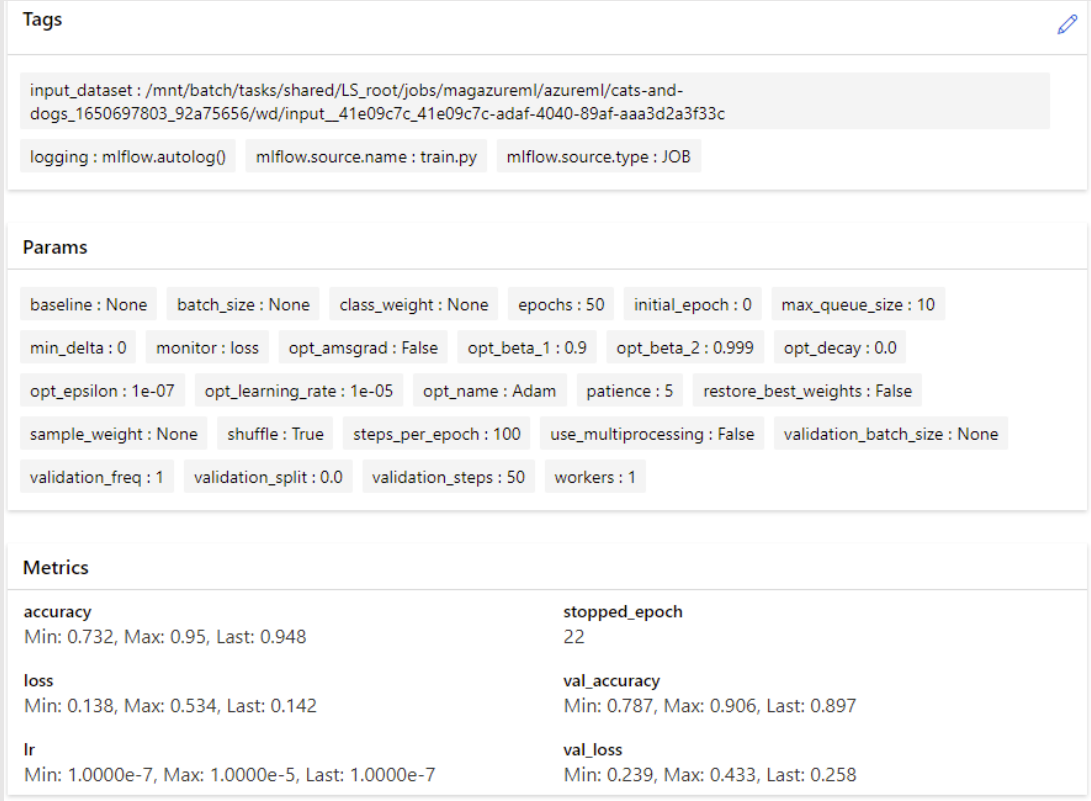


Logging to Experiments

- Mlflow library is already installed when running in compute instances
- Automatic logging allows you to log metrics, parameters, and models without the need for explicit log statements.
- Call mlflow.autolog() before your training code. Enables auto logging for each supported library as soon as it is imported.
- Automatic Logging supports most popular ML libraries, including: TensorFlow and Keras, Scikit-learn, others.
- Use mlflow API for custom/explicit logging

Auto logging for keras

- Metrics:
  - Training loss, validation loss, early stopping metrics
- Params:
  - fit() and fit_generator() params
  - Learning rate, epsilon, optimizer name
- Artifacts
  - Model summary on training start
  - MLFlow model on training end
  - TensorBoard logs
- https://mlflow.org/docs/latest/tracking.html#tensorflow-and-keras



Logging data to Runs using MLFlow

API	Description	Example
mlflow.log_metric mlflow.log_metrics	Logs a metric for a run. Metrics are key-value pair that records a single float measure. A metric can be logged several times. The MLflow Backend keeps track of historical metric values.	mlflow.log_metric("mse", 250.0) metrics = {"mse": 250.0, "rmse": 50.0} mlflow.log_metrics(metrics)
mlflow.log_param mlflow.log_params	Logs a parameter for a run. Examples are params and hyperparams used for ML training. A param is a key-value pair. For a run, a single parameter is allowed to be logged only once.	mlflow.log_param("learning_rate", 0.01) params = {"learning_rate": 0.01, "n_estimators": 10} mlflow.log_params(params)
mlflow.set_tag mlflow.set_tags	Sets a tag on a run. Tags are run metadata that can be updated during a run and after a run completes.	mlflow.set_tag("release.version", "2.2.0")

Mlflow logging functions: https://mlflow.org/docs/latest/tracking.html#logging-functions

Loading Model Artefacts from Run

- Retrieve run details from Azure ML:

```
ws = Workspace.from_config()
exp = Experiment(ws, exp_name)
run = Run(exp, run_name)
```

- Access run metrics:

```
run.get_metrics().keys()
max(run.get_metrics()['val_accuracy'])
```

- Access trained model:

```
run.download_files('model’)
```

- Code file: explore_run.ipynb

Model Training

Watching
a model train

Watching
a model train

Executing Model Training

Interactive Development

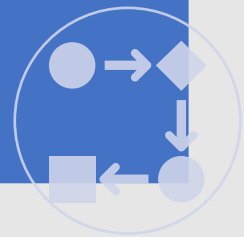
- Build and debug code
- Use tools such as VS Code or Notebooks
- Local development or remote on Compute Instances

Submit Jobs

- Submit a python script as job for long running / unattended execution
- examine logs and results after completion
- Target execution to compute instances or clusters

ML Pipelines

- Multiple steps, e.g.: data prep, feature engineering, modeling and validation
- Reusable and scalable when building MLOPS
- Execute on variety of compute options: CI, cluster, Spark



You could be running on remote compute in any of the three model training approaches

Training on Remote Compute

Execution Environment

- Containerisation is utilised heavily to ensure environment requirements and dependencies are present for job execution
- Select from preconfigured environment, or create a custom env specification (including all python libraries or other programs)
  - AzureML-tensorflow-2.4-ubuntu18.04-py37-cuda11-gpu
- Training script and other code dependencies must be packaged and specified in the job configuration

Data Access

- Data can be presented as mount points on the file system of the training docker container
- Training script must accept command line arguments, argparser is a great utility to work with python arguments
- When job is setup, file datasets are mounted and path to mounted file is passed in to the training script as a command line argument

Exercise: Training Script

Refactor training script:

- Training script: refactor code to run in a training python script
- Utilise arguments, usually using argparser in python, to control parameters of a training job
  - File path to input data set is passed in as a command line argument
  - Other parameters, like training parameters could also be parameterised this way

- Solution file: src/train.py

Exercise: Submit Training Job

Job configuration process:

- Specify target Compute: compute instance or cluster
- Execution environment: select from curated env with required dependencies
- Data access: load dataset as mount point and pass the path in command line argument to the training script
- Submit run: set experiment name and submit run configuration

- Solution File: 05_submit_job.ipynb

yellow_hat_4hqqswpm

Refresh | Connect to compute | Resubmit | Cancel | Delete

Details | Metrics | Images | Child runs | Outputs + logs | Snapshot | Explanations (preview) | Fairness (preview) | Monitoring (preview)

Properties

Status
Completed

Created
Apr 23, 2022 5:10 PM

Started
Apr 23, 2022 5:12 PM

Duration
10m 55.58s

Compute duration
10m 55.58s

Run ID
cats-and-dogs_1650697803_92a75656

Script name
train.py

Created by
Mutaz Abu Ghazaleh

Input datasets
Input name: input__41e09c7c, Dataset: cats_dogs:1

Output datasets
None

Environment
AzureML-tensorflow-2.4-ubuntu18.04-py37-cuda11-gpu:34

Arguments
--input_dataset
DatasetConsumptionConfig:input__41e09c7c

Registered models
None

Git repository
https://github.com/mutazag/azureml-samples

Git branch
main

Git commit
42f4bc762492d8afe3788b2477974a325d87840b

See all properties
Raw JSON

See YAML job definition
Job YAML

Compute

Target
gpucluster

Instance count
1

Compute type
amlcompute

Tags

input_dataset : /mnt/batch/tasks/shared/LS_root/jobs/magazureml/azureml/cats-and-dogs_1650697803_92a75656/wd/input__41e09c7c_41e09c7c-adaf-4040-89af-aaa3d2a3f33c

logging : mlflow.autolog() | mlflow.source.name : train.py | mlflow.source.type : JOB

Params

baseline : None | batch_size : None | class_weight : None | epochs : 50 | initial_epoch : 0 | max_queue_size : 10 | min_delta : 0 | monitor : loss | opt_amsgrad : False | opt_beta_1 : 0.9 | opt_beta_2 : 0.999 | opt_decay : 0.0 | opt_epsilon : 1e-07 | opt_learning_rate : 1e-05 | opt_name : Adam | patience : 5 | restore_best_weights : False | sample_weight : None | shuffle : True | steps_per_epoch : 100 | use_multiprocessing : False | validation_batch_size : None | validation_freq : 1 | validation_split : 0.0 | validation_steps : 50 | workers : 1

Metrics

accuracy
Min: 0.732, Max: 0.95, Last: 0.948

stopped_epoch
22

loss
Min: 0.138, Max: 0.534, Last: 0.142

val_accuracy
Min: 0.787, Max: 0.906, Last: 0.897

lr
Min: 1.0000e-7, Max: 1.0000e-5, Last: 1.0000e-7

val_loss
Min: 0.239, Max: 0.433, Last: 0.258

Description

Click edit icon to add a description