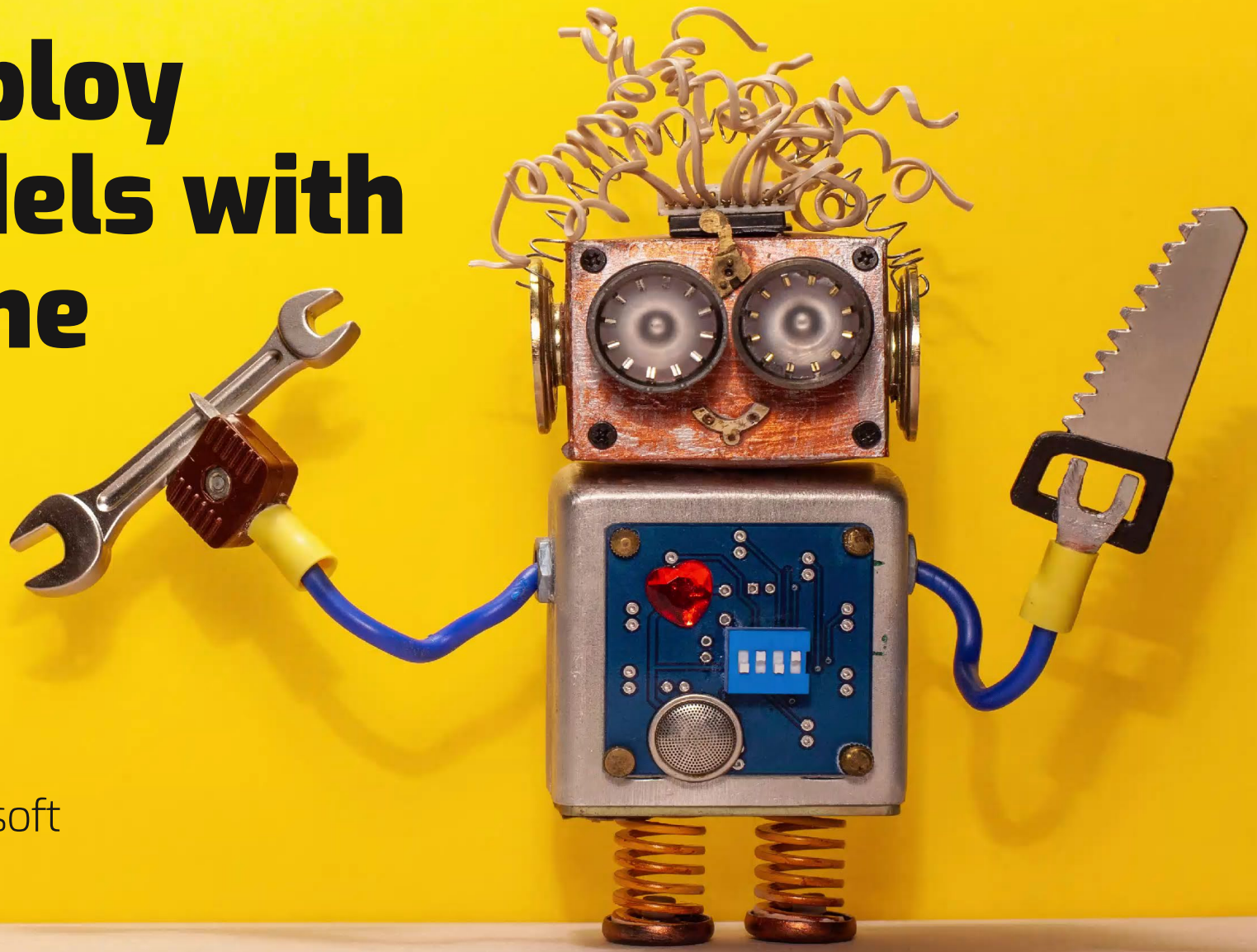
# Build and deploy PyTorch models with Azure Machine Learning



**Henk Boelman**
Cloud Advocate @ Microsoft

HenkBoelman.com

@hboelman

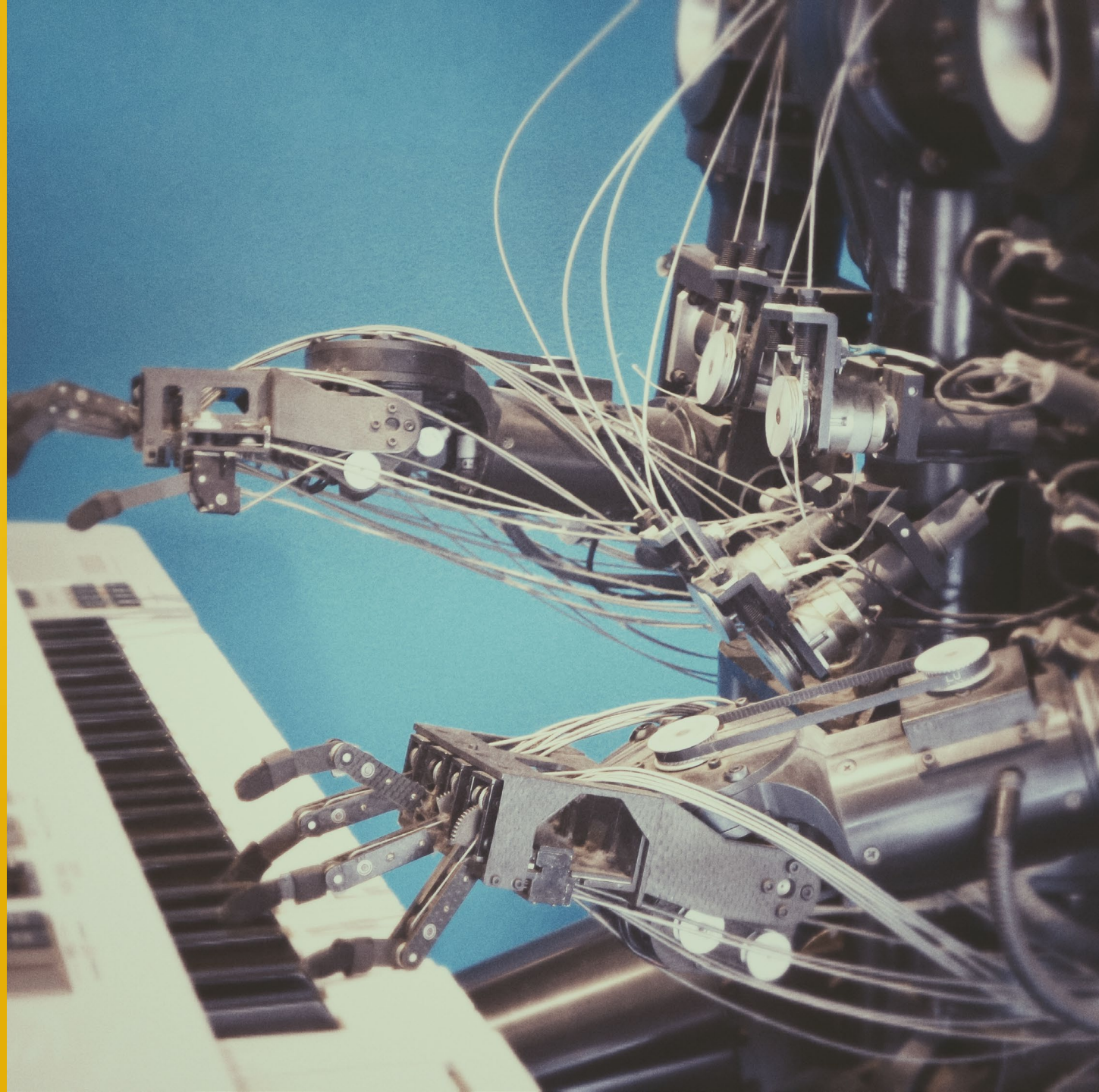
# Agenda

Machine Learning on Azure

Simpson classification using Azure Machine Learning

# Machine Learning

Ability to learn without being explicitly programmed.



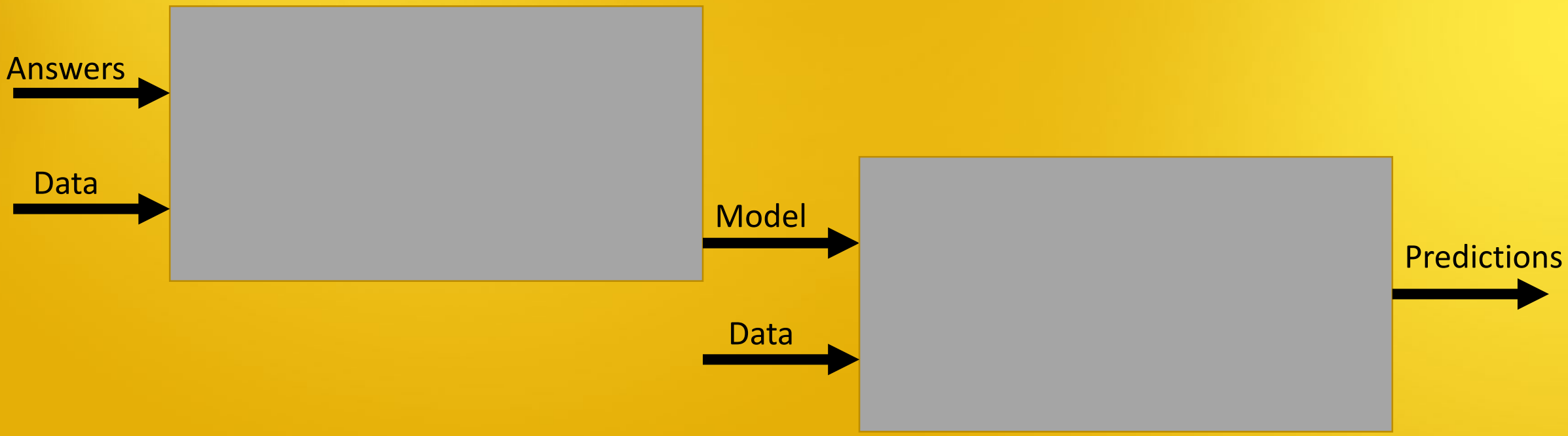
# Programming

# Machine Learning

# Machine Learning

# Machine Learning

# Machine Learning

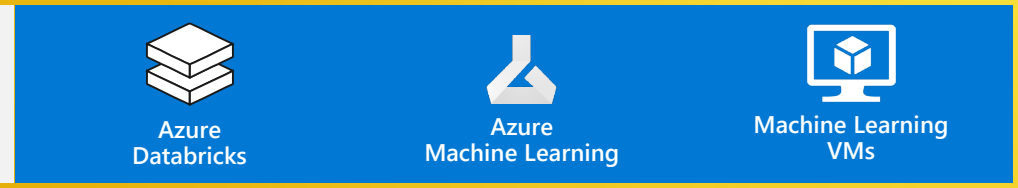


# Machine Learning on Azure

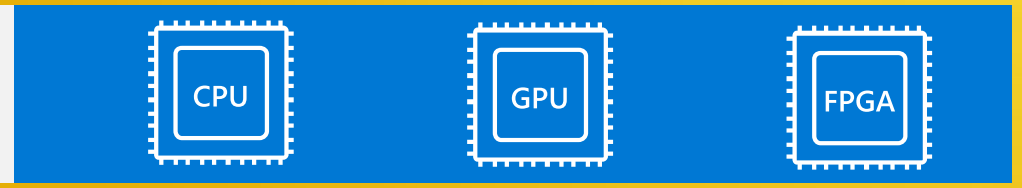
## Sophisticated pretrained models
To simplify solution development


## Popular frameworks
To build advanced deep learning solutions

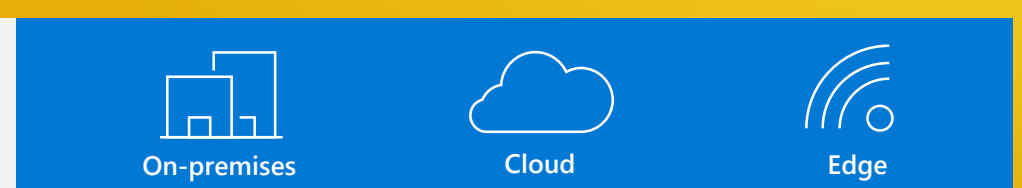

## Productive services
To empower data science and development teams




## Powerful infrastructure
To accelerate deep learning




## Flexible deployment
To deploy and manage models on intelligent cloud and edge



# Cognitive Services:

Pre-Trained models in the cloud and on the edge

Vision

Speech

Language

Knowledge


LEVEL UP!

# Azure Machine Learning studio

A fully-managed cloud service that enables you to easily build, deploy, and share predictive analytics solutions.


# Is it Marge or Homer or .. or ..?

# Simpson Lego dataset

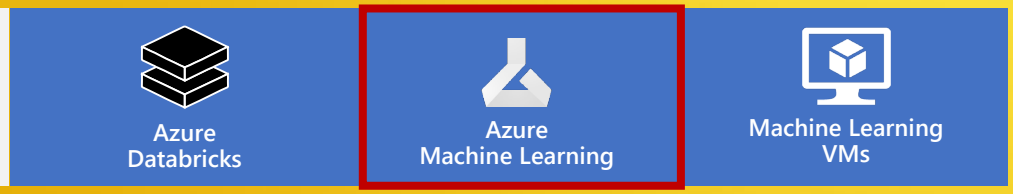

https://github.com/hnky/dataset-lego-figures

# Machine Learning on Azure

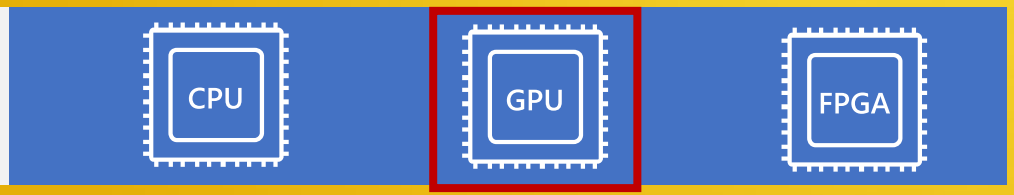
## Sophisticated pretrained models
To simplify solution development


## Popular frameworks
To build advanced deep learning solutions

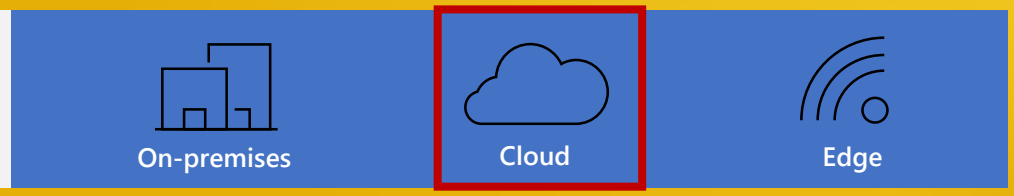

## Productive services
To empower data science and development teams




## Powerful infrastructure
To accelerate deep learning




## Flexible deployment
To deploy and manage models on intelligent cloud and edge



Data Scientist


Favorite Tools

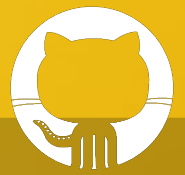
Version Control

Azure Machine Learning Service

Model management


Developer

Favorite Tools



Version Control

Build Pipeline

Azure Container Registry

Operations

Deploy Pipeline

Azure Kubernetes Service

# What is Azure Machine Learning Service?

Set of Azure Cloud Services

Python SDK

That enables you to:

- ✓ Prepare Data
- ✓ Build Models
- ✓ Train Models
- ✓ Manage Models
- ✓ Track Experiments
- ✓ Deploy Models

**Prepare**

your environment

**Experiment**

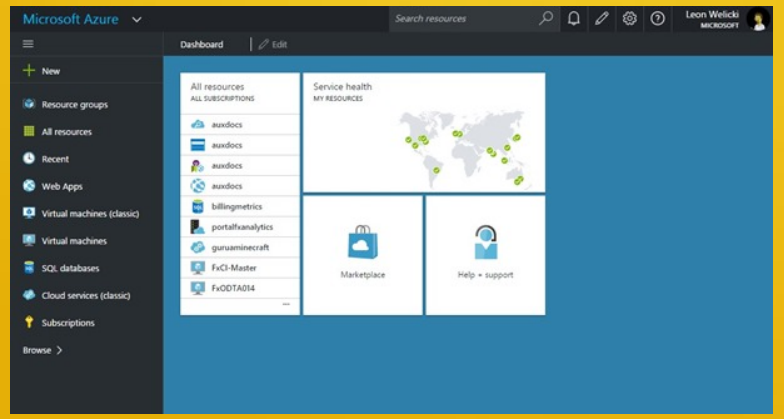
with your model & data

**Deploy**

Your model into production

# Step 1: **Prepare your environment**

# Setup your environment




Azure Portal

VS Code

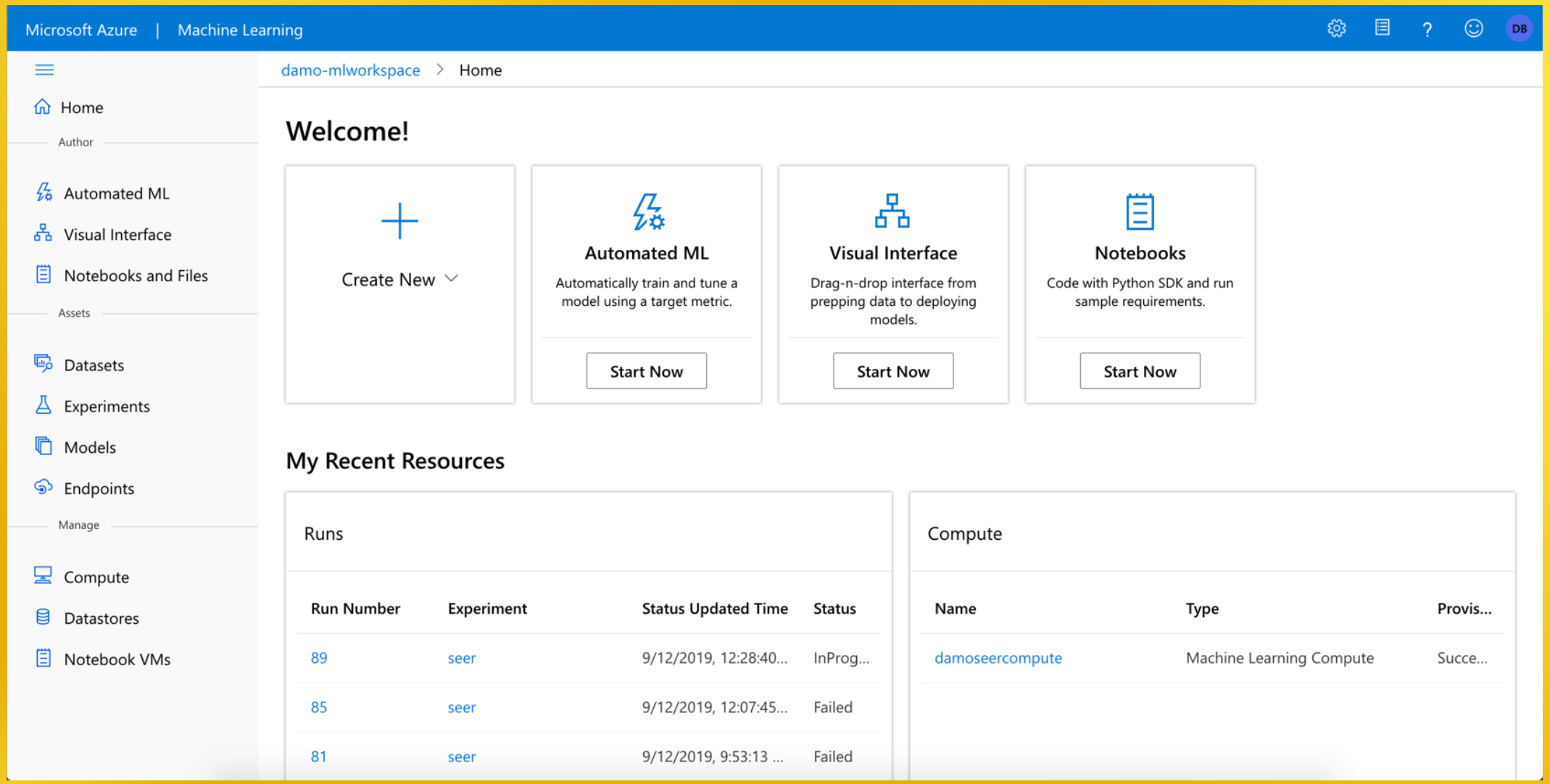
Python SDK

# Create a workspace

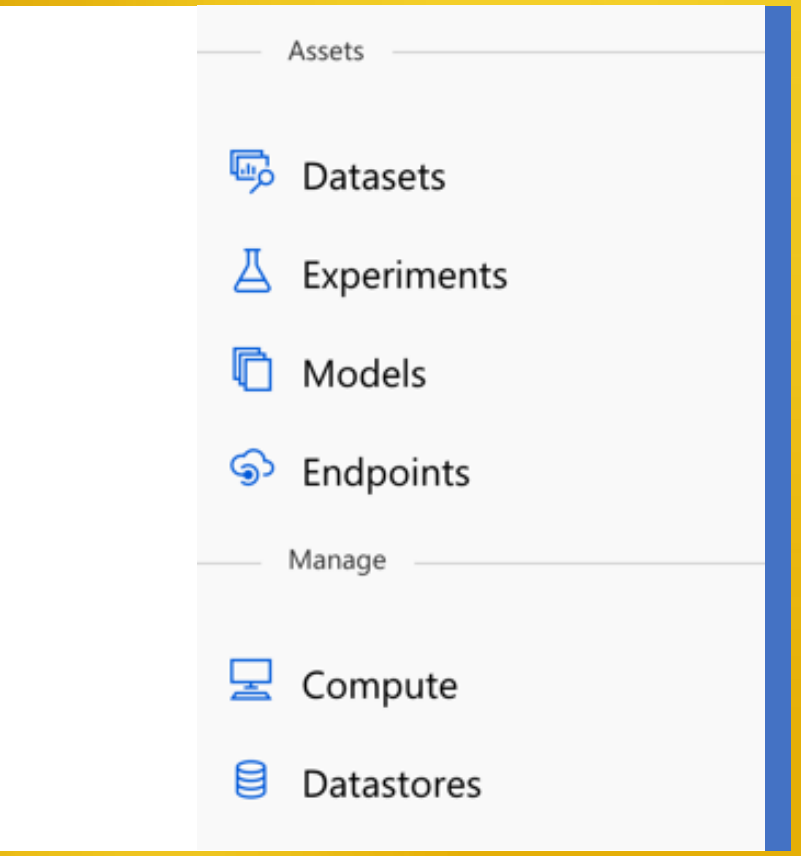
```
ws = Workspace.create(
             name='<NAME>',
             subscription_id='<SUBSCRIPTION ID>',
             resource_group='<RESOURCE GROUP>',
             location='westeurope')

ws.write_config()

ws = Workspace.from_config()
```

Create a workspace

# ml.azure.com


Microsoft Azure | Machine Learning
DB
damo-mlworkspace > Home
Home
Author
Automated ML
Visual Interface
Notebooks and Files
Assets
Datasets
Experiments
Models
Endpoints
Manage
Compute
Datastores
Notebook VMs
Welcome!
Create New
Automated ML
Automatically train and tune a model using a target metric.
Start Now
Visual Interface
Drag-n-drop interface from prepping data to deploying models.
Start Now
Notebooks
Code with Python SDK and run sample requirements.
Start Now
My Recent Resources
Runs
Run Number
Experiment
Status Updated Time
Status
89
seer
9/12/2019, 12:28:40...
InProg...
85
seer
9/12/2019, 12:07:45...
Failed
81
seer
9/12/2019, 9:53:13 ...
Failed
Compute
Name
Type
Provis...
damoseercompute
Machine Learning Compute
Succe...

# Azure Machine Learning Service




**Datasets** – registered, known data sets

**Experiments** – Training runs

**Models** – Registered, versioned models

**Endpoints:**

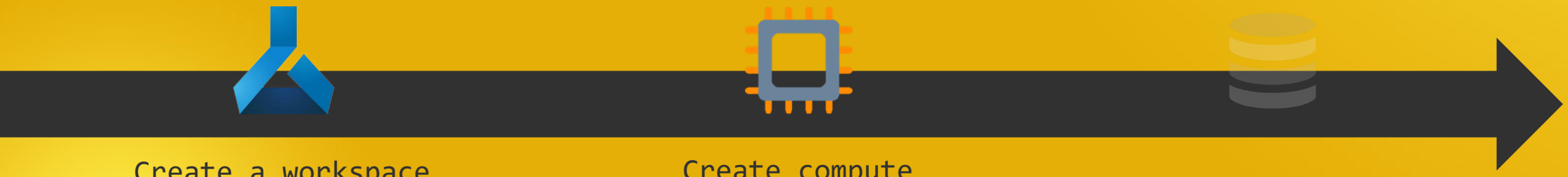
- **Real-time Endpoints** – Deployed model endpoints
- **Pipeline Endpoints** – Training workflows

**Compute** – Managed compute

**Datastores** – Connections to data

# Create Compute

```
cfg = AmlCompute.provisioning_configuration(
            vm_size='STANDARD_NC6',
            min_nodes=1,
            max_nodes=6)

cc = ComputeTarget.create(ws, '<NAME>', cfg)
```



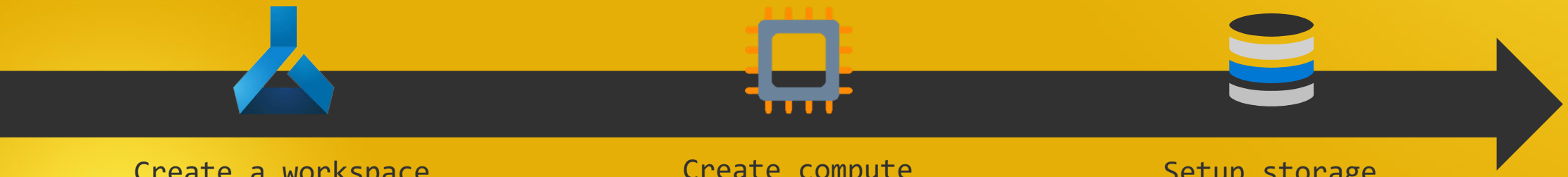
Create a workspace

Create compute

# Attach storage

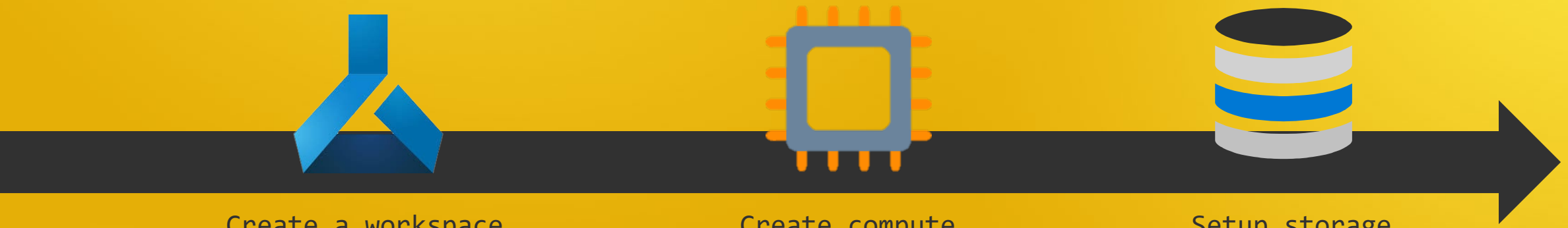
```
ds = ws.get_default_datastore()

ds = Datastore.register_azure_blob_container(
                workspace = ws,
                datastore_name = '<your datastore name>’,
                container_name = '<your blob container name>’,
                account_name = '<your storage account name>’,
                account_key = '<your storage account key>’
)
```

# Demo: **Setup your workspace**



Step 1

# Prepare your environment



Create a workspace

Create compute

Setup storage

# Step 2: **Experiment with your model & data**

# Create an experiment

```
exp = Experiment(workspace=ws, name="<ExperimentName>")
```

Create an Experiment

# Create a training file

```
model=Sequential()
model.add(Conv2D(kernel_size=(3,3),filters=3,input_shape=(128,128,1),activation="relu"))
model.add(Conv2D(kernel_size=(3,3),filters=10,activation="relu",padding="same"))
model.add(MaxPooling2D(pool_size=(2,2),strides=(2,2)))
model.add(Conv2D(kernel_size=(3,3),filters=3,activation="relu"))
model.add(Conv2D(kernel_size=(5,5),filters=5,activation="relu"))
model.add(MaxPooling2D(pool_size=(3,3),strides=(2,2)))
model.add(Conv2D(kernel_size=(2,2),strides=(2,2),filters=10))
model.add(Flatten())
model.add(Dropout(0.3))
model.add(Dense(100,activation="sigmoid"))
model.add(Dense(1,activation="sigmoid"))
model.summary()

model.compile(optimizer="adadelta",loss="binary_crossentropy",metrics=["accuracy"])

history = model.fit(data1, labels, validation_split=0.2, epochs=30, batch_size=20, callbacks=[historyLoss])
```

Create an Experiment

Create a training file

# Create an Environment

**Curated enviroment**

```
curated_env_name = 'AzureML-PyTorch-1.6-GPU'
pytorch_env = Environment.get(workspace=ws, name=curated_env_name)
pytorch_env = pytorch_env.clone(new_name='pytorch-1.6-gpu’)
```

**Custom environment**

```
env = Environment('pytorch-1.6-gpu')
cd = CondaDependencies.create(
        pip_packages=['azureml-dataprep[pandas,fuse]', 'azureml-defaults'],
        conda_packages=['pytorch','torchvision'])
env.python.conda_dependencies = cd
env.docker.enabled = True
```

Create an Experiment

Create a training file

Create an ScriptRunConfig

# Create an ScriptRunConfig

```
args = ['--data-folder', simpsons_ds.as_named_input('simpsons').as_mount(),
        '--num-epochs', 10]

project_folder = "./trainingscripts“

config = ScriptRunConfig(
    source_directory = project_folder,
    script = 'train.py',
    compute_target=compute_target,
    environment = pytorch_env,
    arguments=args,
)
```

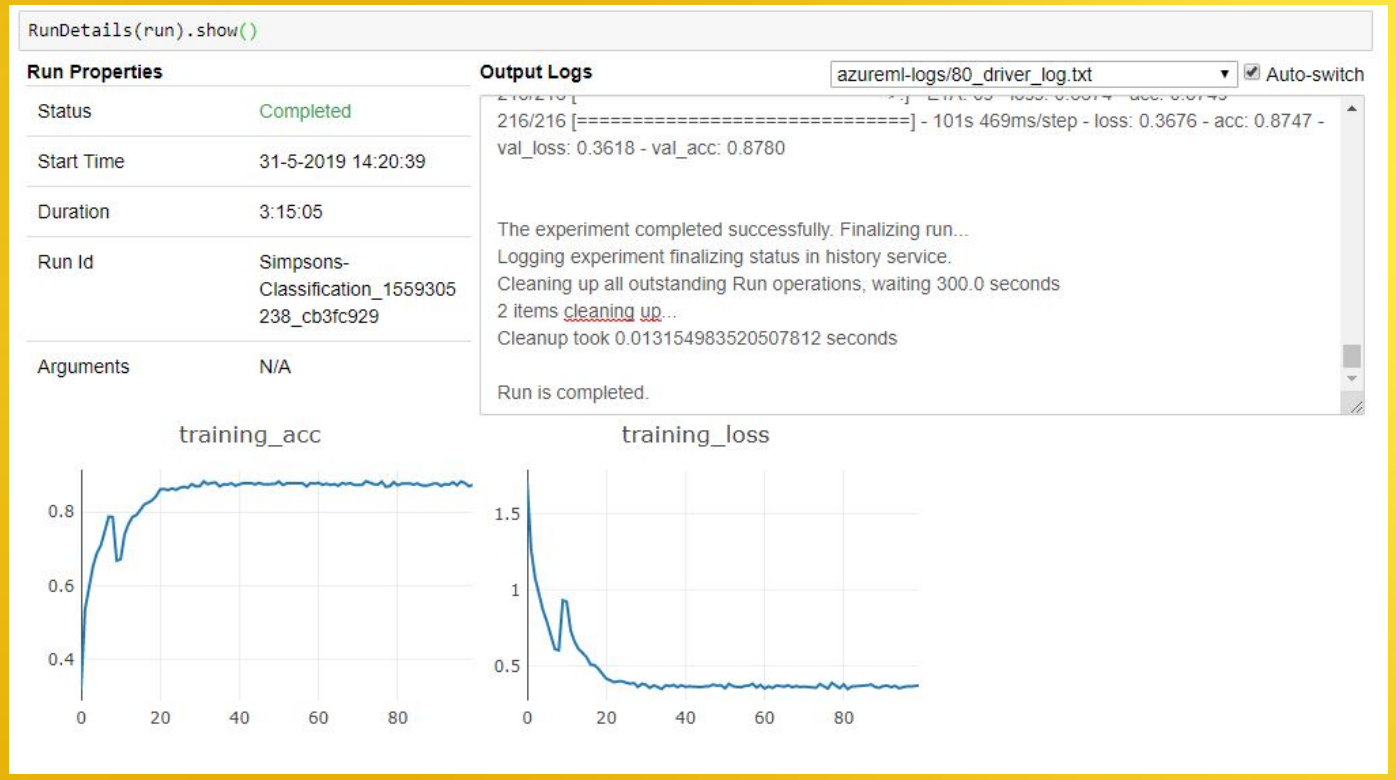
Create an Experiment

Create a training file

Create an ScriptRunConfig

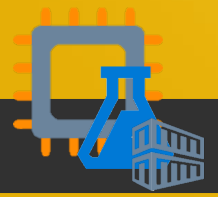
# Submit the experiment to the cluster

```
run = exp.submit(estimator)
RunDetails(run).show()
```









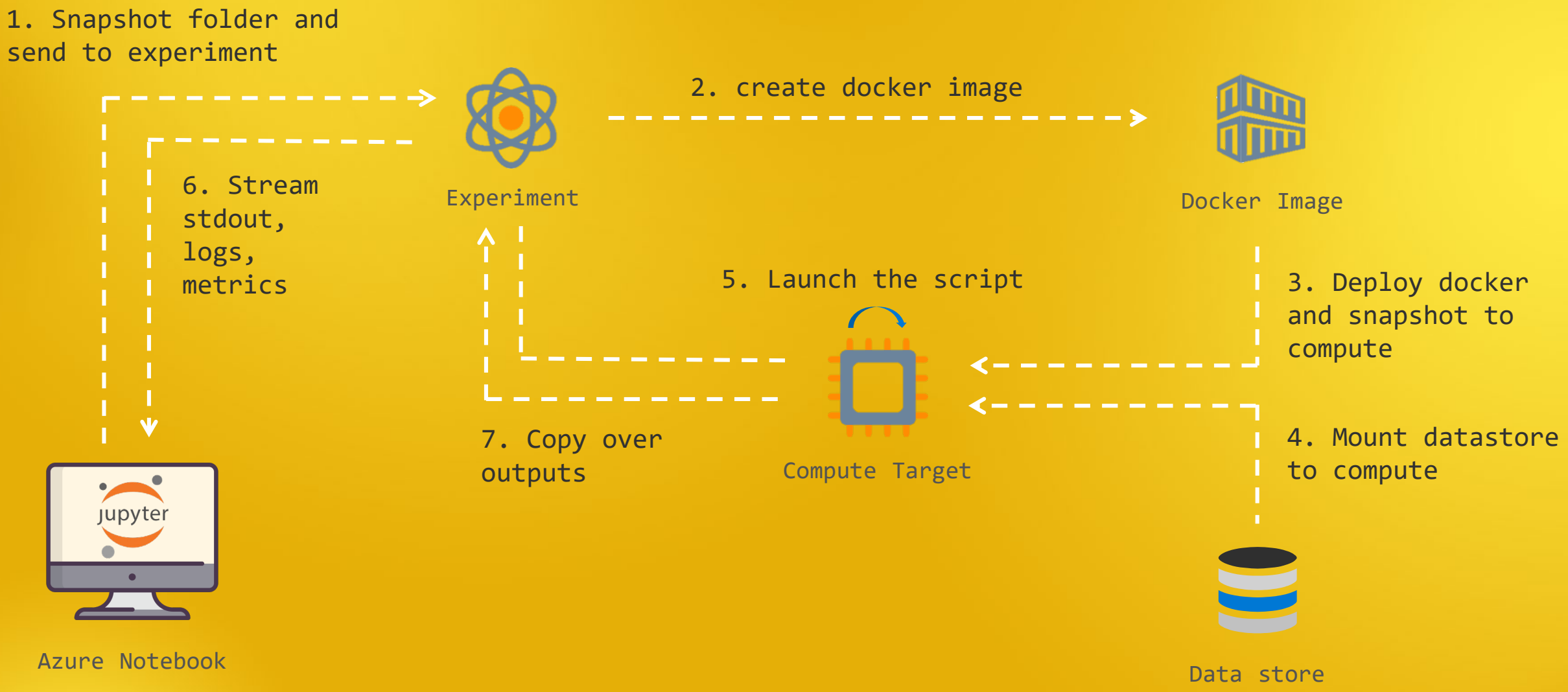

Create an Experiment

Create a training file

Create an ScriptRunConfig

Submit to the AI cluster

# Demo: **Creating and run an experiment**


1. Snapshot folder and send to experiment
Experiment
2. create docker image
Docker Image
6. Stream stdout, logs, metrics
5. Launch the script
3. Deploy docker and snapshot to compute
7. Copy over outputs
Compute Target
4. Mount datastore to compute
jupyter
Azure Notebook
Data store

# Register the model

```
model = run.register_model(
                    model_name='SimpsonsAI',
                    model_path='outputs')
```

# Demo: **Register and test the model**

Step 2

# Experiment with your model & data

Create an Experiment

Create a training file

Create an ScriptRunConfig

Submit to the AI cluster

Register the model

# Step 3: **Deploy your model into production**

# AMLS to deploy

The Model

Score.py

Environment file

Docker Image

# Score.py

```
%%writefile score.py
from azureml.core.model import Model

def init():
      model_root = Model.get_model_path('MyModel’)
      loaded_model = model_from_json(loaded_model_json)
      loaded_model.load_weights(model_file_h5)

def run(raw_data):
      url = json.loads(raw_data)['url’]
      image_data = cv2.resize(image_data,(96,96))
      predicted_labels = loaded_model.predict(data1)
      return json.dumps(predicted_labels)
```

# Environment File

```
from azureml.core.runconfig import CondaDependencies

cd = CondaDependencies.create()

cd.add_conda_package('keras==2.2.2')

cd.add_conda_package('opencv')

cd.add_tensorflow_conda_package()

cd.save_to_file(base_directory='./', conda_file_path='myenv.yml')
```

```
# Conda environment specification. The dependencies defined in this file will
# be automatically provisioned for runs with userManagedDependencies=False.

# Details about the Conda environment file format:
# https://conda.io/docs/user-guide/tasks/manage-environments.html#create-env-file-manually

name: project_environment
dependencies:
  # The python interpreter version.
  # Currently Azure ML only supports 3.5.2 and later.
- python=3.6.2

- pip:
  - azureml-defaults==1.0.2
- numpy
- keras==2.2.2
- opencv
- tensorflow=1.10.0

```

# Inference config

```
inference_config = InferenceConfig(
    runtime= "python",
    entry_script="score.py",
    conda_file="myenv.yml"
)
```

# Deployment using AMLS

Deployment

Azure Container Instance (ACI)

Azure Kubernetes Service (AKS)

# Deploy to ACI

```
aciconfig = AciWebservice.deploy_configuration(
                    cpu_cores = 1,
                    memory_gb = 2)

service = Model.deploy(workspace=ws,
                           name='simpsons-aci',
                           models=[model],
                           inference_config=inference_config,
                           deployment_config=aciconfig)
```

# Deploy to AKS

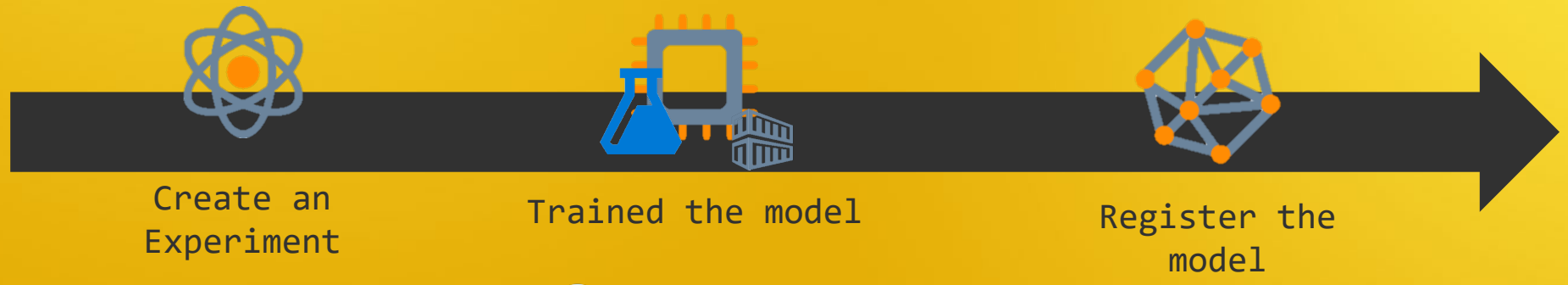
```
aks_target = AksCompute(ws,"AI-AKS-DEMO")

deployment_config = AksWebservice.deploy_configuration(
    cpu_cores = 1,
    memory_gb = 1)

service = Model.deploy(workspace=ws,
                       name="simpsons-ailive",
                       models=[model],
                       inference_config=inference_config,
                       deployment_config=deployment_config,
                       deployment_target=aks_target)

service.wait_for_deployment(show_output = True)
```

# Demo: **Deploy to ACI**

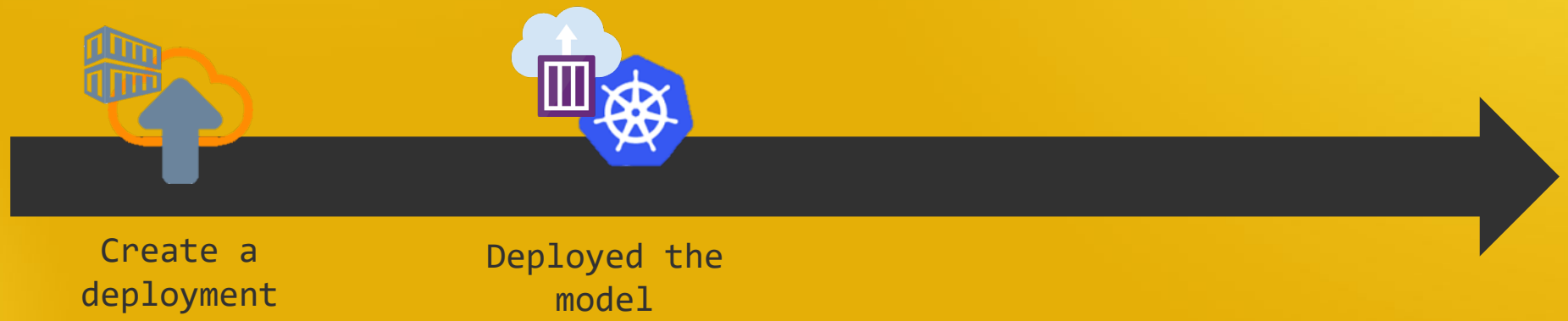
# Recap

**Prepare**
your environment


**Experiment**
with your model & data




**Deploy**
your model



Code on: **aka.ms/amls-pytorch**

# Thank you!

Read more on: **aka.ms/amls-pytorch**

@hboelman

github.com/hnky

henkboelman.com