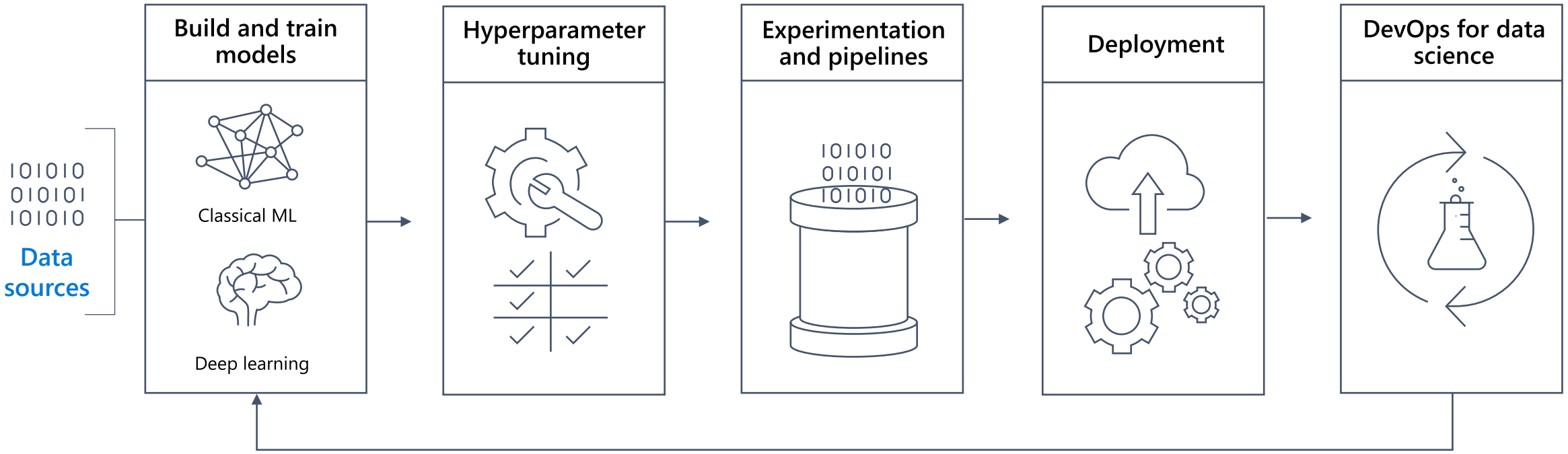
De l'entrainement au déploiement en un script avec le SDK azureml

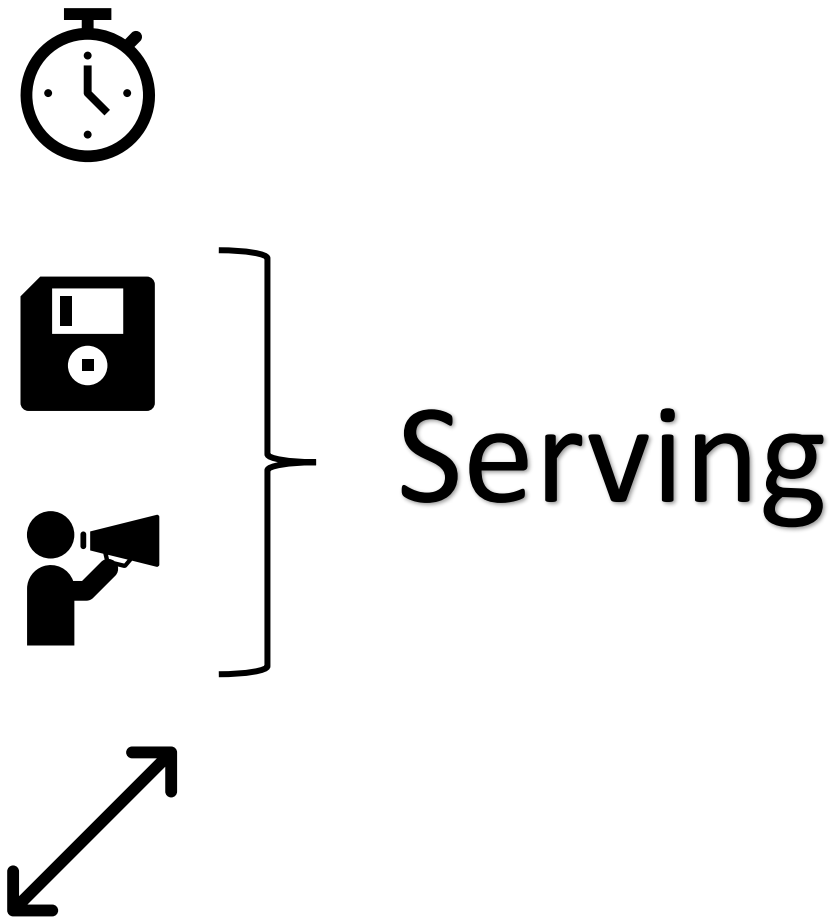
Paul PETON – Microsoft AI MVP – Consultant @ AZEO Nantes
Twitter : @paulpeton
https://www.linkedin.com/in/paul-peton-datascience
https://github.com/methodidacte/
Blog : http://methodidacte.org

Building blocks for a Data Science Project



What do we need for production ?

- A scheduler to plan
  - Data cleansing
  - Training / re-training of the model
  - The forecast calculation (in batch mode)
- A storage system to archive models
  - Per algorithm, per version, per training dataset
  - In a serialized (not proprietary) binary format
- A tool for exposing the model
  - Via API REST (diagnostic language)
  - Secure access
- Resources that can be deployed at scale
  - With the help of the containers
  - In the Cloud



Azure Machine Learning Service

Set of Azure Cloud Services

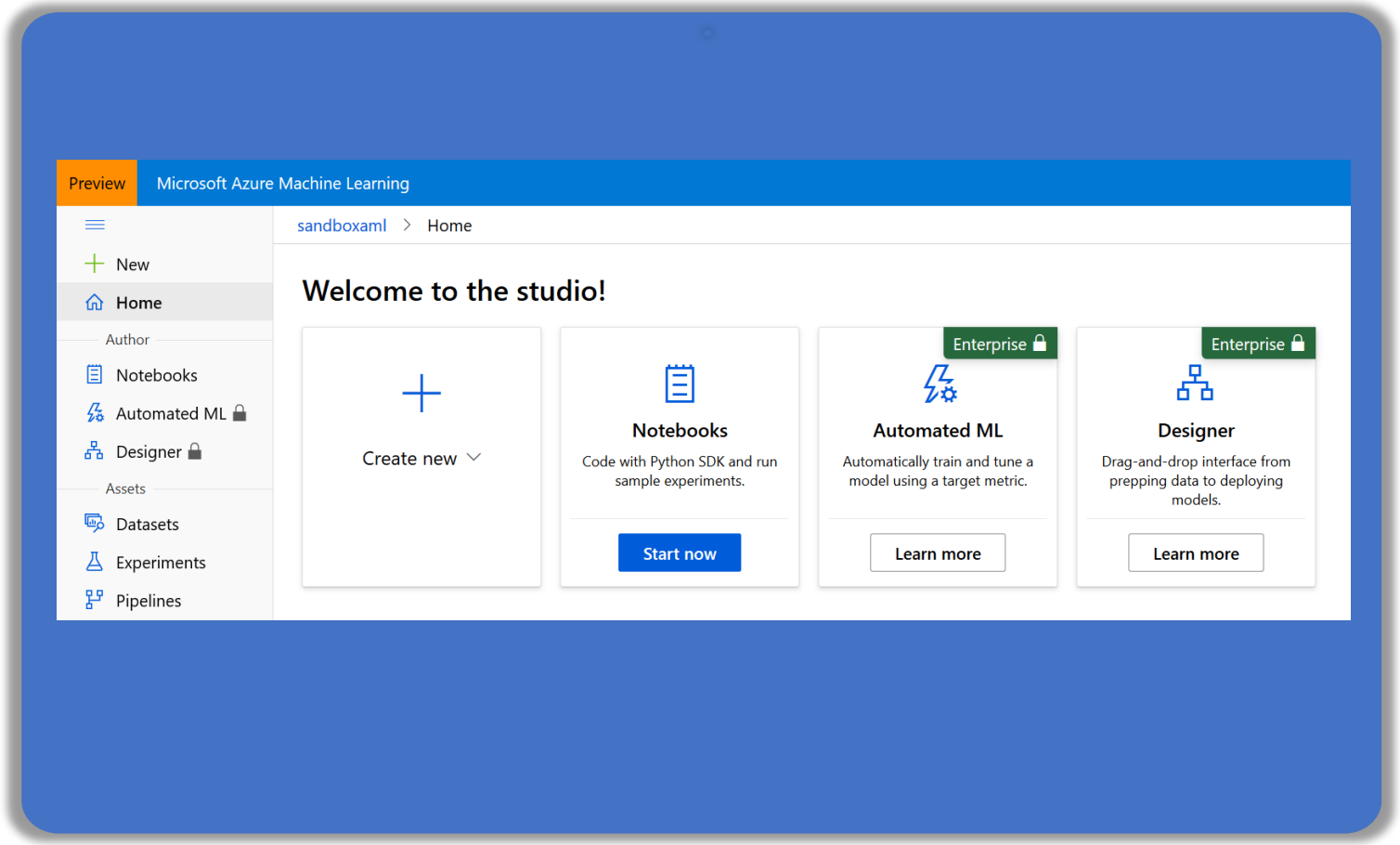
Python SDK & R

That enables you to:

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

Azure Machine Learning

The “new” Studio



Azure Machine Learning components

Experience
SDK, Notebooks, Drag-n-drop, Wizard

MLOps
Reproducible, Automatable, GitHub, CLI, REST

Datasets
Profiling, Drift, Labeling

Training
Experiments, Runs

Model Registry
Models, Images

Inferencing
Batch, Realtime

Compute
Jobs, Clusters, Instances

Azure IoT Edge
Security, Mgmt., Deployment

Cloud
CPU, GPU, FPGA

Edge
CPU, GPU, NPU

Supported Azure Storage services

- Azure Blob Container
- Azure File Share
- Azure Data Lake
- Azure Data Lake Gen2
- Azure SQL Database
- Azure Database for PostgreSQL
- Azure Database for MySQL
- Databricks File System

Azure Python SDK

- Set of libraries that facilitate access to :
  - Management components (Virtual Machine, Cluster, Image…)
  - Runtime components (ServiceBus using HTTP, Batch, Monitor…)

- Official GitHub repository :
https://github.com/Azure/azure-sdk-for-python
- The full list of available packages and their latest version :
https://docs.microsoft.com/fr-fr/python/api/overview/azure/?view=azure-python

- Installation :

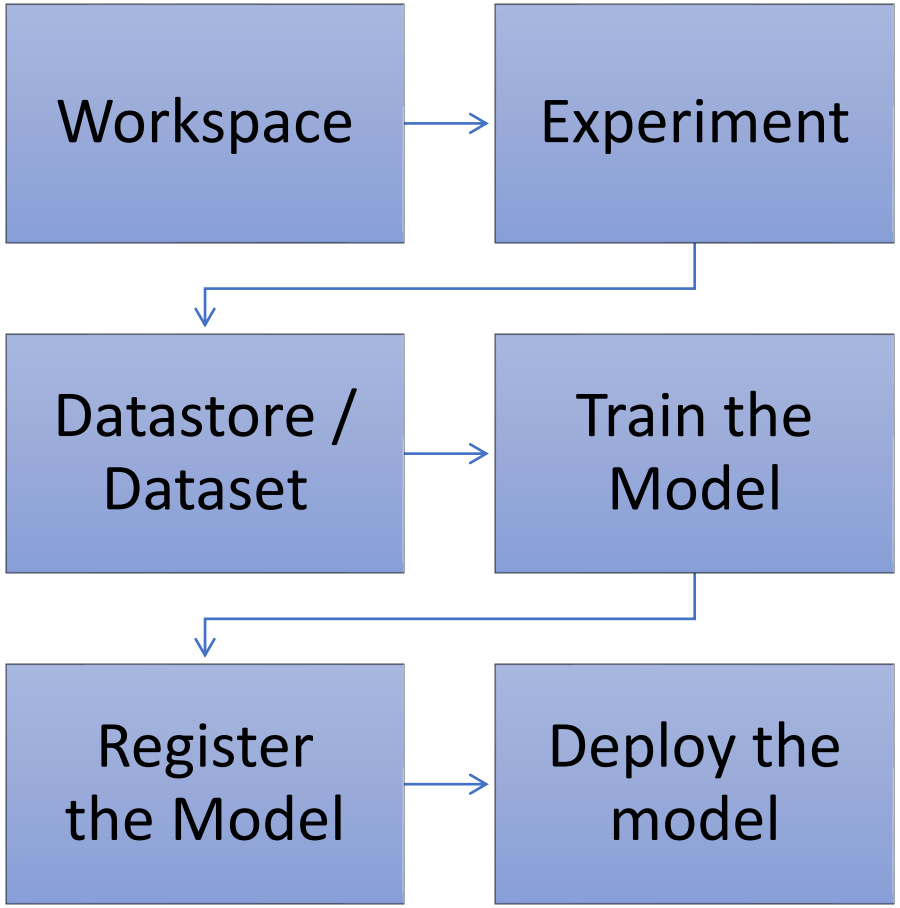
```
!pip install --upgrade azureml-sdk
```

- Or clone th GtiHub reporsitory :

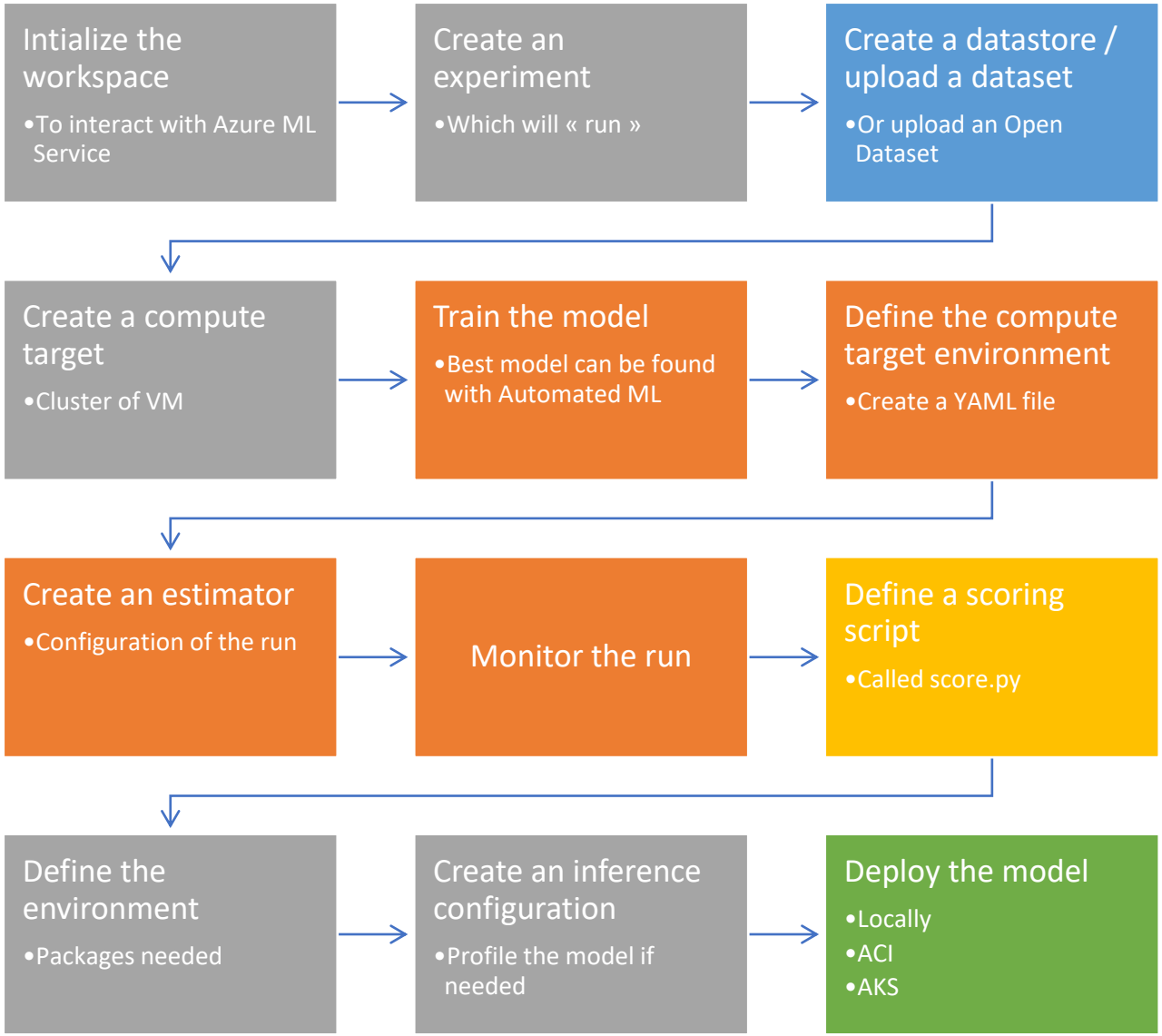
```
git clone git://github.com/Azure/azure-sdk-for-python.git
cd azure-sdk-for-python
python setup.py install
```

Main Objects

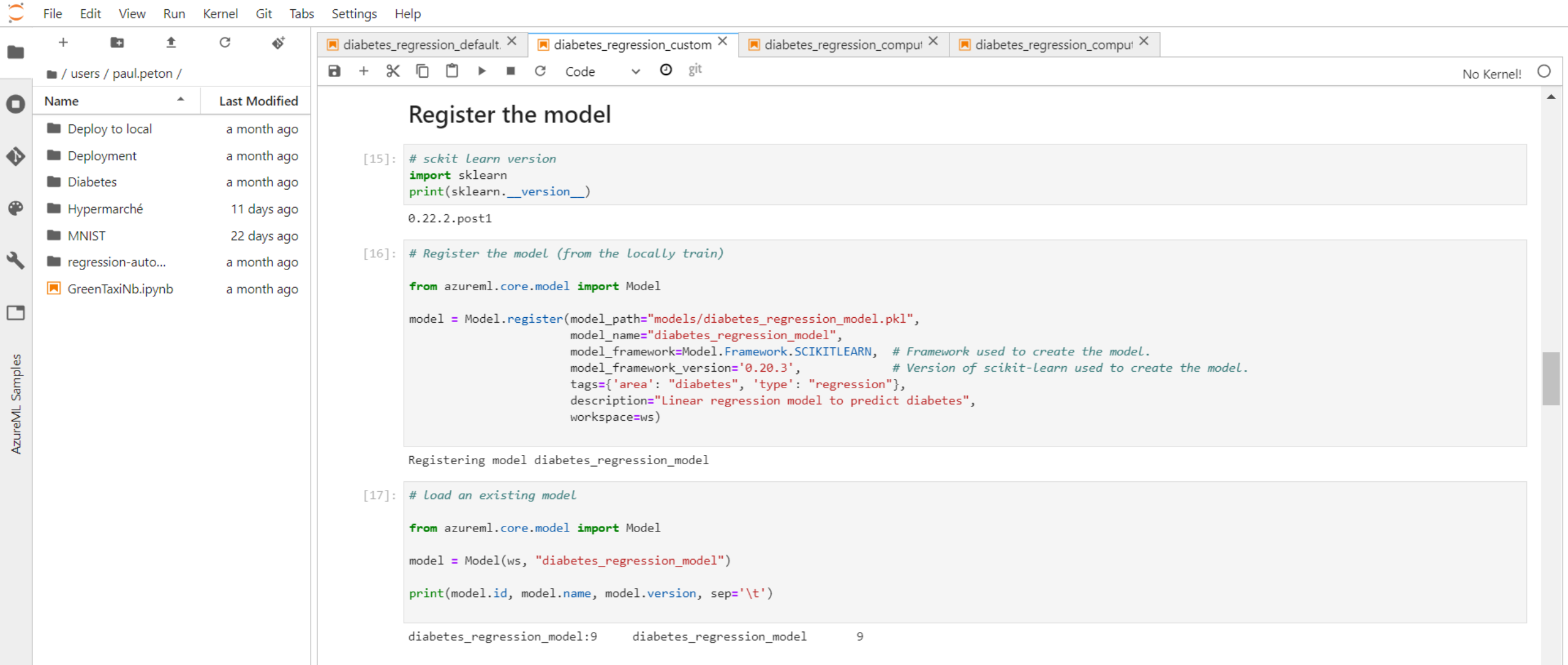
- Workspace
- Inside the Workspace
  - Datastore & Dataset
  - Compute target
  - Experiment
    - Run
  - Pipeline
  - Model
    - Environment
    - Estimator
  - Inference
  - Endpoint



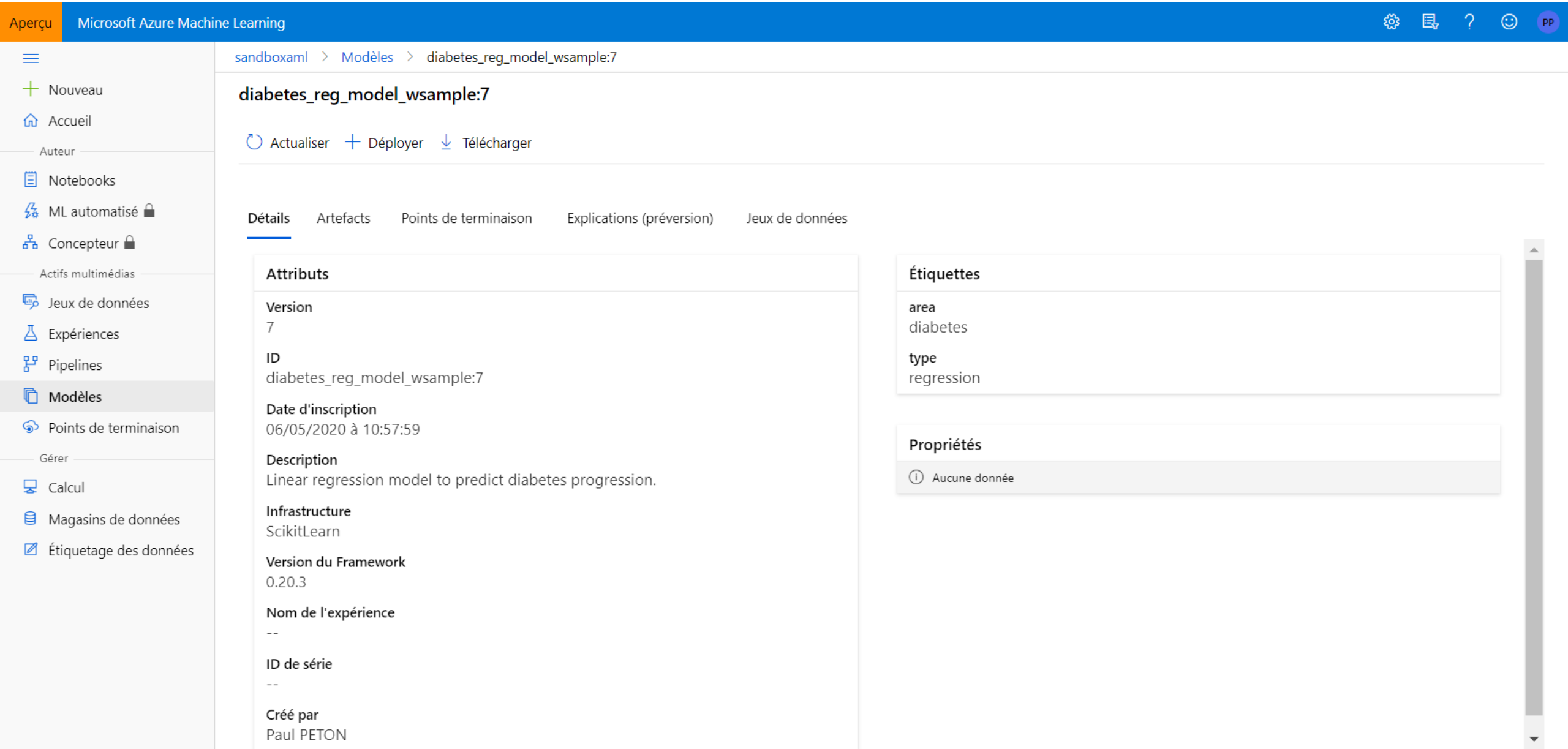
ML « **remote** » workflow with Python SDK



Register model from a notebook



Models versioning (User Interface : « studio »)



Deploy a model on Azure Container Instance

File Edit View Run Kernel Git Tabs Settings Help

/ users / paul.peton /

Name	Last Modified
Deploy to local	a month ago
Deployment	a month ago
Diabetes	a month ago
Hypermarché	11 days ago
MNIST	22 days ago
regression-auto...	a month ago
GreenTaxiNb.ipynb	a month ago

AzureML Samples

diabetes_regression_default | diabetes_regression_custom | diabetes_regression_compu | diabetes_regression_compu

Code git No Kernel!

Define the (inference) environement

[22]:
```
from azureml.core import Environment
from azureml.core.conda_dependencies import CondaDependencies


environment = Environment('my-sklearn-environment')
environment.python.conda_dependencies = CondaDependencies.create(pip_packages=[
    'azureml-defaults',
    'inference-schema[numpy-support]',
    'joblib',
    'numpy',
    'scikit-learn'
])
```

Define a inference configuration

[23]:
```
from azureml.core.model import InferenceConfig
inference_config = InferenceConfig(entry_script='score.py', environment=environment)
```

Deploy in a custom environment

[24]:
```
from azureml.core import Webservice
from azureml.core.webservice import AciWebservice
from azureml.exceptions import WebserviceException


service_name = 'diabetes-custom-service'
# Remove any existing service under the same name.
try:
    Webservice(ws, service_name).delete()
except WebserviceException:
    pass

aci_config = AciWebservice.deploy_configuration(cpu_cores=1, memory_gb=1)

service = Model.deploy(workspace=ws,
                       name=service_name,
                       models=[model],
                       inference_config=inference_config,
                       deployment_config=aci_config)
```