



# Azure Machine Learning & Databricks : friends or foe ?

*Wenesday, the 1st of june, 2022*

**Microsoft®**  
Most Valuable  
Professional



**Award Categories**  
AI, Data Platform

**First year awarded:**  
2018

**Number of MVP Awards:**  
4

# Speaker : Paul PETON

- Manager Analytics & Data Science @AVANADE Nantes
  - *We are hiring*
- Microsoft MVP Data Platform & AI since 2018
- Specialised in the industrialisation of ML models on Azure
- Blog : <https://methodidacte.org>
- LinkedIn : <https://www.linkedin.com/in/paul-peton-datascience/>
- Twitter : @paulpeton

I can do everything  
with only one tool

In French :

*“touche à tout, bon à rien”*

In English :

*“all-purpose, no-purpose”*



# Agenda : the right tool at the right place

- « Ops » for Machine Learning
- DEMO : a taxi story...
- Azure ML + Databricks
- MLOps stack
- Azure ML + Synapse Analytics ?
- #finops : do it cheaper



# A definition of « Ops » for Machine Learning



Automatisation

**“End to End”**: from raw data to predictive use of the model



Continuous X

**Test developments** as much as possible: unit tests, non-regression tests, etc.



Versioning

**Deploy** versioned developments in a continuous and automated way



Testing

**Monitor:**



Reproductibilité

- storage / calculation / exposure resources
- serving: response time AND response "quality"



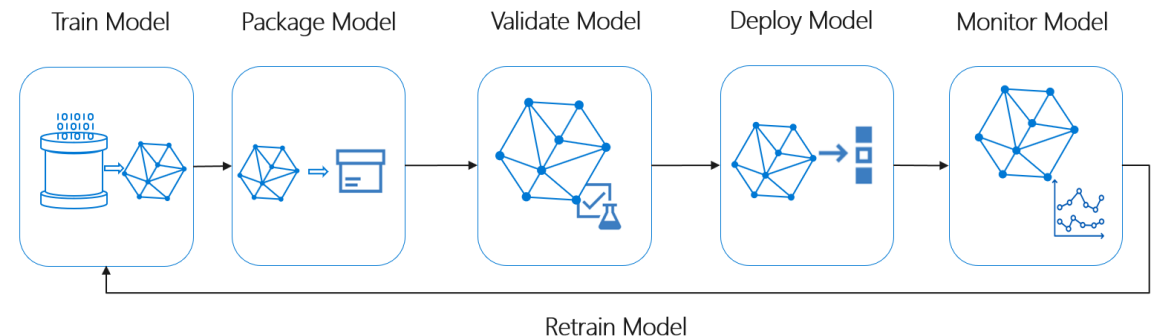
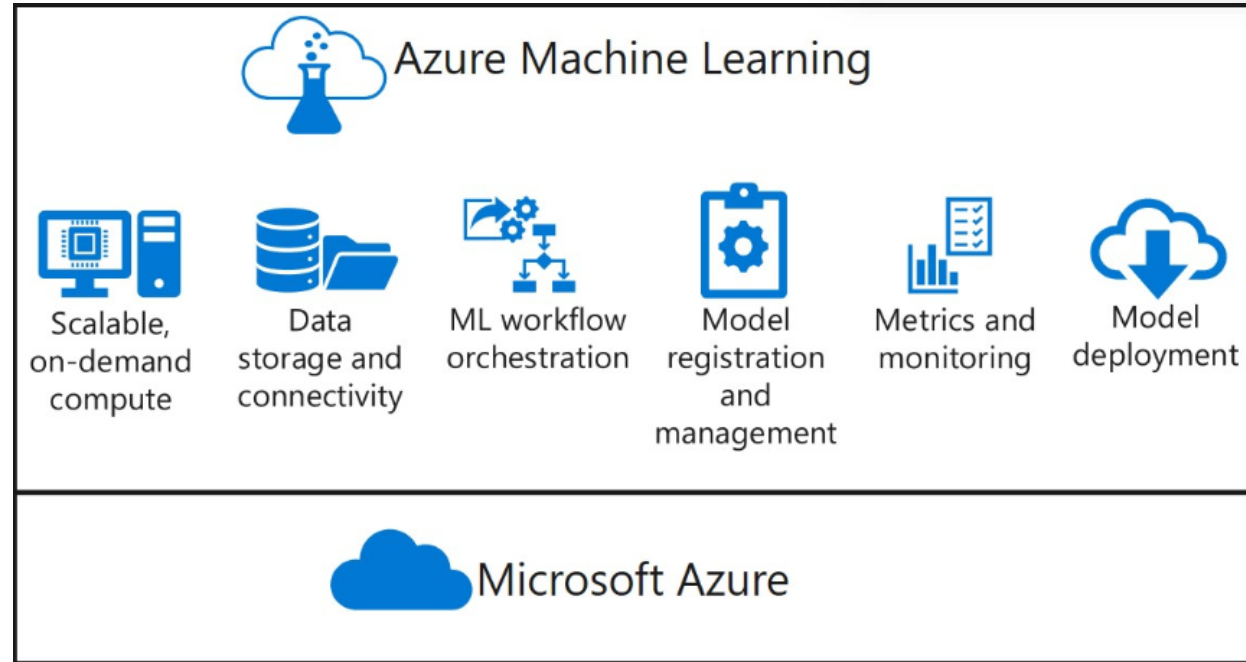
Monitoring

**React to drifts**: data drift, concept drift, etc.

# Azure Machine Learning in a few words

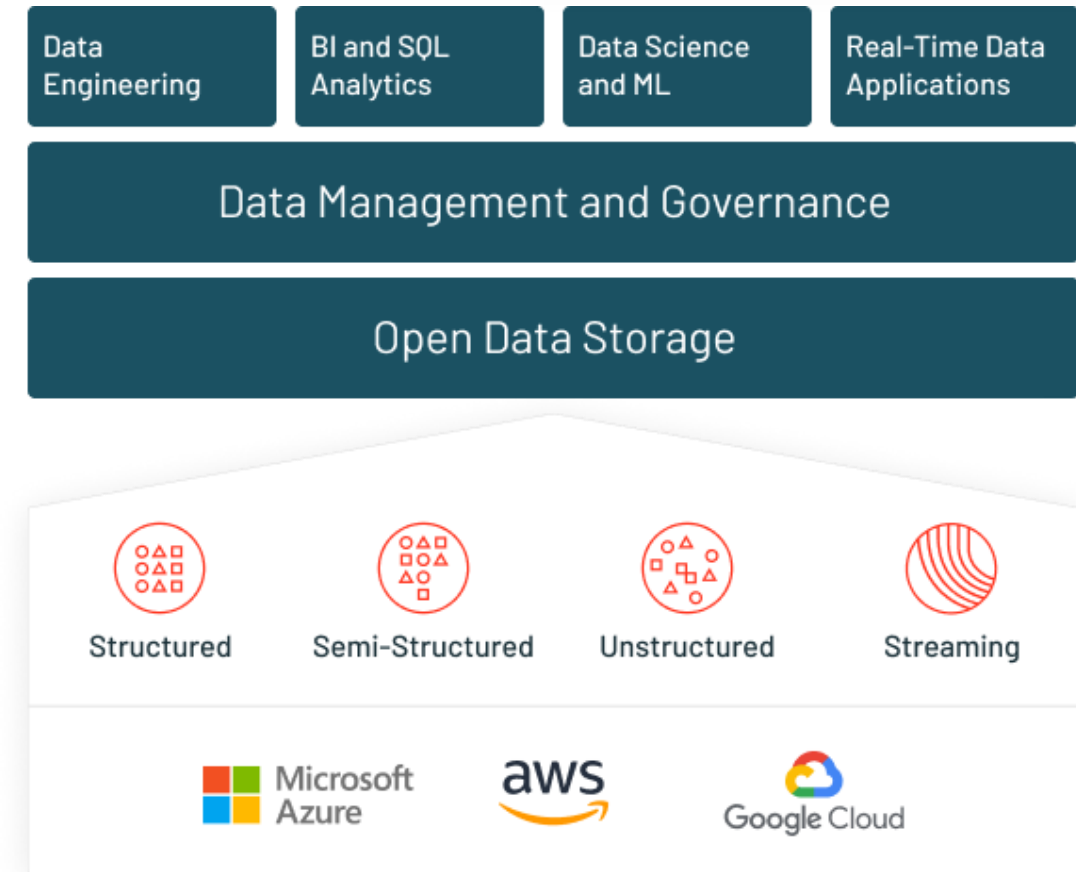


- Portal for
  - Citizen Data Scientists
    - no code, autoML, data labelling
  - Data Engineers
    - code + logs
- MLOps tool
  - From dev to production
  - Store, compute, versioning, expose



# Azure Databricks in a few words

- Available on the three major public clouds : AWS, Azure, GCP
- Product vision : « *unified analytics platform* »
- Define the « **Lakehouse** » approach
- Integrate the Open Source tool **MLFlow**
- Partnership with the tool **dbt**
- Open Source languages (Scala, Python, R) and SQL
- Ability to '**scale out**' with the power of Spark, a distributed computing engine



# A taxi story...

---

- Yellow Taxi dataset
- Data from 2018 to march 2022
  - [Web site](#)
  - Monthly Parquet files
- Stored in a Azure Data Lake gen2
- 252 094 475 rows
- Objective :
  - Load the dataset in memory
  - Train a model of “fare\_amount”
  - Register the model
  - Prediction in batch or real-time





# First try : notebook in Azure ML



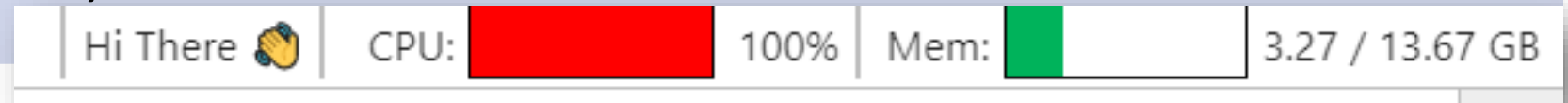
Define a Dataset in Azure ML



Compute instance : Standard\_DS3\_v2      4 cores, 14 GB RAM, 28 GB disk



Jupyter : memory error => restart Kernel



We need to scale up !

It's not a shame  
But it can be done differently

# Read data as a Pandas dataframe

A screenshot of the Azure ML portal interface. The left sidebar shows a navigation menu with options like 'Répertoire par défaut', 'New', 'Home', 'Notebooks', 'Automated ML', 'Designer', 'Assets', 'Data', 'Jobs', 'Components', 'Pipelines', 'Environments', 'Models', and 'Endpoints'. The main panel displays the 'yellow\_taxi\_parquet' dataset page. It includes tabs for 'Details', 'Consume', 'Explore', and 'Models'. Below these are buttons for 'New version', 'Refresh', 'Generate profile', and 'Unregister'. A 'Sample usage' section contains a code block with Python code for connecting to the workspace and reading the dataset. The line `dataset.to_pandas_dataframe()` is highlighted with a red rectangle.

Répertoire par défaut > centralizedaml > Data > yellow\_taxi\_parquet

yellow\_taxi\_parquet Version 1 (latest) ▾

Details Consume Explore Models

New version ▾ Refresh ▶ Generate profile 🔗 Unregister

Sample usage 📄

```
# azureml-core of version 1.0.72 or higher is required
# azureml-dataprep[pandas] of version 1.1.34 or higher is required
from azureml.core import Workspace, Dataset

subscription_id = 'f80606e5-788f-4dc3-a9ea-2eb9a7836082'
resource_group = 'rg-centralized-registry'
workspace_name = 'centralizedaml'

workspace = Workspace(subscription_id, resource_group, workspace_name)

dataset = Dataset.get_by_name(workspace, name='yellow_taxi_parquet')
dataset.to_pandas_dataframe()
```

pandas provides data structures for in-memory analytics

# Scale up : the only option ?

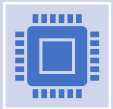


Instance	vCPU	RAM (GiB)	Temporary storage	Pay as you go (\$)
			(GiB)	
DS1 v2	1	3,5	7	<b>91,98</b>
DS2 v2	2	7	14	<b>183,96</b>
DS3 v2	4	14	28	<b>367,92</b>
DS4 v2	8	28	56	<b>735,84</b>
DS5 v2	16	56	112	<b>1471,68</b>

# And what about a compute target ?



Define a compute cluster in Azure ML and run code remotely



VM size : Standard\_DS3\_v2

Maximum number of nodes : 4



Run error

✓✗ UserError

```
{'code': ExecutionFailed, 'message': [{"exit_code": -1, "error_message": "Possibly out of memory. Execution failed with error: INFO: current python version is 3.8.13 (default, Mar 28 2022, 11:38:47) \n[GCC
```



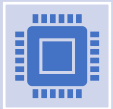
Can't distribute Pandas dataframe

We need a Spark context !

# Going to Azure Databricks...



Start a Databricks cluster, authenticate to Azure ML workspace



VM size : Standard\_DS3\_v2

1 worker and 1 driver node



`dataset.to_spark_dataframe()`

Should work with Scala 2.11, issue on Scala 2.12

```
⊕ java.lang.NoSuchMethodError: okhttp3.HttpUrl.get(Ljava/lang/String;)Lokhttp3/HttpUrl;
```



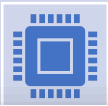
Can't read data from Datastore

Forget about Azure ML Datasets !

# ... trying to distribute ML training



Try with Scikit-Learn which needs a Numpy Array



Same issue as Pandas dataframe

We need to use Spark libraries

`org.apache.spark.SparkException: Job aborted due to stage failure: Task 7 in stage 12.0 failed 4 times, most recent failure: Lost task 7.3 in stage 12.0 (TID 82) (10.139.64.6 executor 0): java.lang.UnsupportedOperationException: org.apache.parquet.column.values.dictionary.PlainValuesDictionary$PlainDoubleDictionary`



MLLib is obsolete

Spark ML could be an option



And what about SynapseML ?

Let's see this wonderful library !

# Synapse Machine Learning library

SynapseML (previously MMLSpark) is an **open source library** to simplify the creation of scalable machine learning pipelines. SynapseML builds **on Apache Spark and SparkML** to enable new kinds of machine learning, analytics, and model deployment workflows.

SynapseML adds many deep learning and data science tools to the Spark ecosystem, including **seamless integration of Spark Machine Learning pipelines** with :

[Open Neural Network Exchange \(ONNX\)](#)












[LightGBM](#)

[The Cognitive Services](#)

[Vowpal Wabbit](#)

[OpenCV.](#)



			
Vowpal Wabbit on Spark	The Cognitive Services for Big Data	LightGBM on Spark	Spark Serving
			
HTTP on Spark	ONNX on Spark	Responsible AI	Spark Binding Autogeneration
			
Isolation Forest on Spark	CyberML	Conditional KNN	

# Install SynapseML on Azure Databricks



Install library

Library Source

Upload

DBFS/ADLS

PyPI

Maven

CRAN

Workspace

Coordinates

com.microsoft.azure:synapseml\_2.12:0.9.5

Search Packages

Repository ?

Optional

Exclusions

Dependencies to exclude (log4j:log4j,junit:junit)

Cancel

Install

SynapseML requires **Scala 2.12**, **Spark 3.2+**, and **Python 3.6+**.





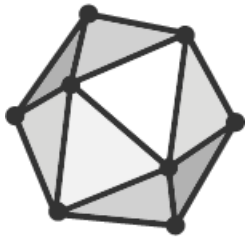
[LightGBM](#) is an open-source, distributed, high-performance gradient boosting (GBDT, GBRT, GBM, or MART) framework. LightGBM is part of Microsoft's [DMTK](#) project.

### **Advantages :**

- Composability : can be incorporated into existing SparkML Pipelines
- Performance : 10-30% faster than SparkML
- Functionality : wide array of tunable parameters
- Cross platform : available on Spark, PySpark, and SparklyR

### **Usages :**

- Classifier
- Regressor including quantile regression
- Ranker



# ONNX, what's that ?!?

- **Open Neural Network Exchange**
- ONNX is an open format built to represent Machine Learning models
  - Community project
  - Interoperability
  - Pre-trained models
    - <https://github.com/onnx/models>
- Converting to ONNX format
  - <http://onnx.ai/sklearn-onnx/index.html>



# Azure Databricks limitations



- MLFlow is too much integrated in the Databricks workspace
  - We don't need a registry per environment
  - Stages of a model can't be **inside** a workspace
  - We need an **external registry**
- Serving
  - without monitoring
  - without advanced authentication
  - without configuration fine tuning
  - with DBU cost

Name	Latest Version	Staging	Production
dev_model	Version 1	Version 1	-
housing_elastic	Version 1	-	-
housing_model	Version 6	Version 6	Version 5

Registered Models > dev\_model

### dev\_model

Details [Serving](#)

Status: ● Pending - Stop Cluster: [mlflow-model-dev\\_model](#)

[Model Versions](#) [Model Events](#) [Cluster Settings](#)

#### Cluster Settings

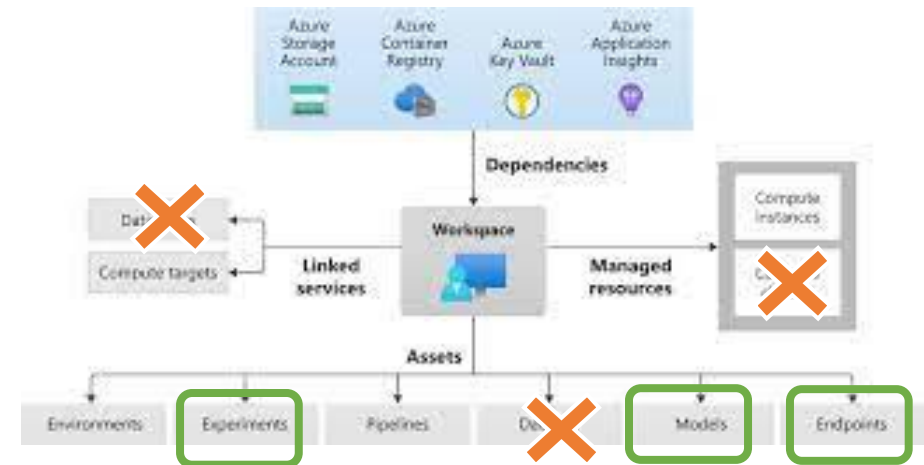
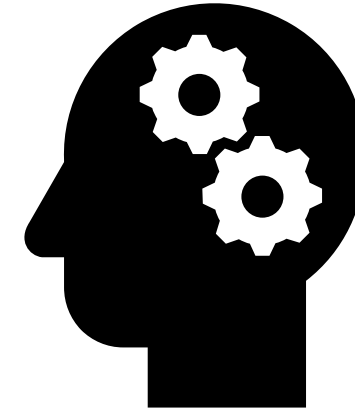
Change the configuration of the cluster used in serving this endpoint.

Instance Type

Standard\_F4 8.0 GB Memory, 4 Cores

# Some conclusions

- At scale or in production, **don't use every feature of Azure ML**
  - Forget about Datastore and Dataset
    - Use Delta format in the Data Lake*
  - Just explore data inside notebooks on compute instance
    - Don't forget refactoring and packaging of the code*
  - Forget about compute cluster
    - Can't really distribute the code*
  - Don't create an Azure ML workspace per environment
- In a « MLOps stack », Azure ML must be « **central** »
  - Logs of experiments
  - Model Tracking
  - Model Registry
  - Serving : deploy predictive endpoints



Let's talk about  
MLOps stack



# What do we need to version in MLOps ?

- Classical development project

{infra as code, app code, CI/CD code} + last data

- Machine Learning project

{data, code, model+requirements, container}

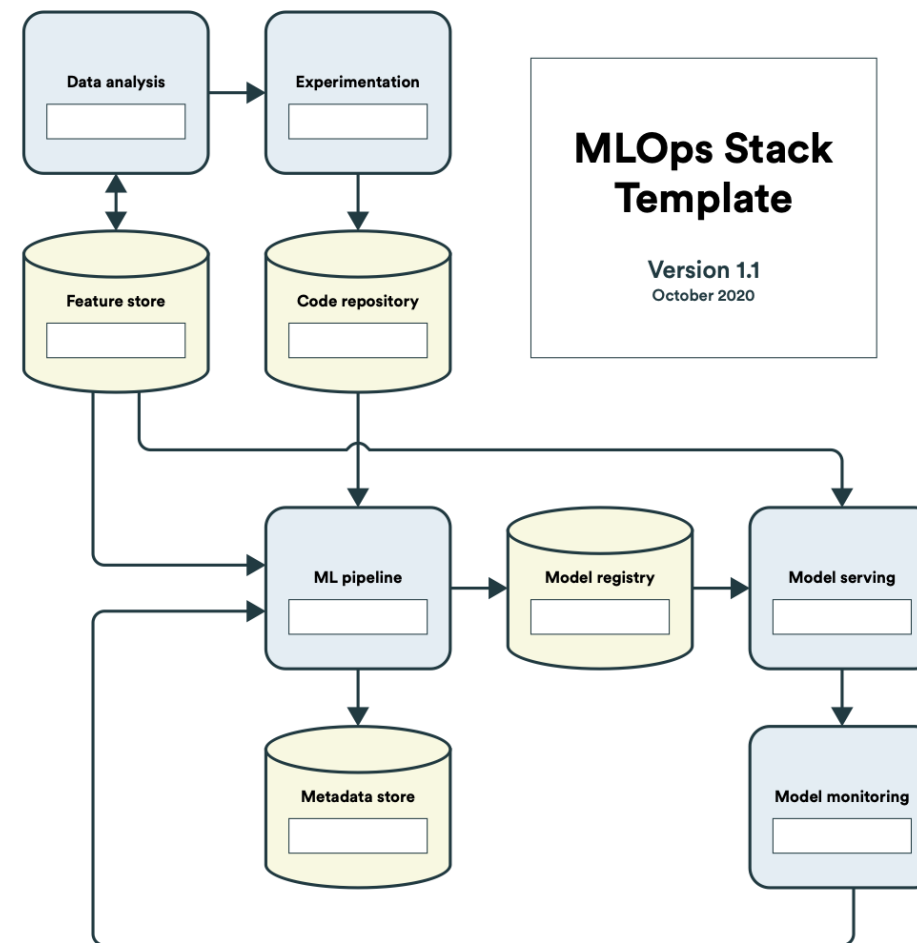
# MLOps : what we need to version

{ (raw / clean) data,  
ML code + tests,  
model + requirements,  
CI/CD code,  
container}

# ML-Ops.org - principles

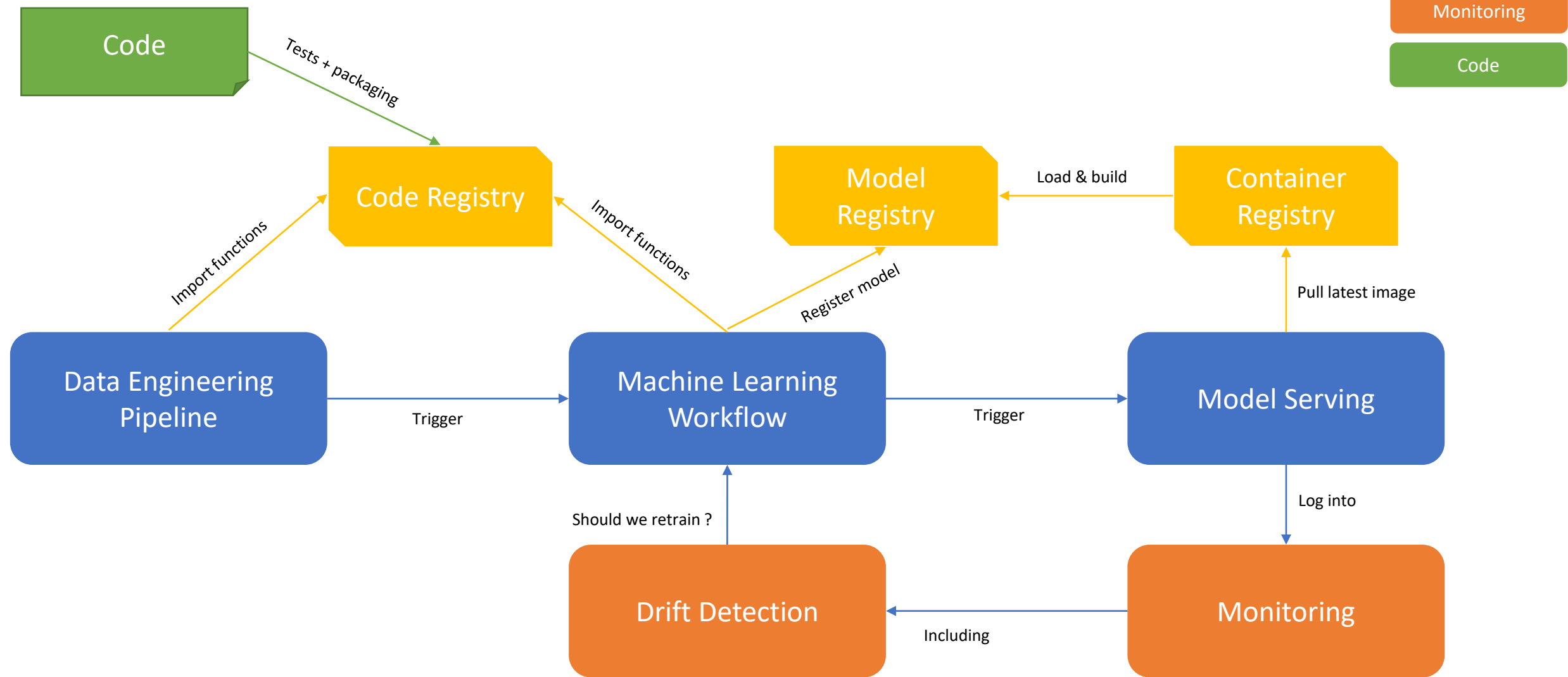
created by Dr. Larysa Visen2geriyeva, Anja Kammer, Isabel Bär, Alexander Kniesz, and Michael Plöd (DDD Advisor)

- The three main « pipelines » :
  - Data Engineering Pipeline
  - Machine Learning Pipeline / Workflow
  - Model Serving
- The concept of « registry » :
  - Storage adapted to the type of deliverable
    - Code
    - Binary (pickle, H5, etc.)
    - Docker image
    - etc.
  - Allows versioning of objects
  - Ensures reproducibility

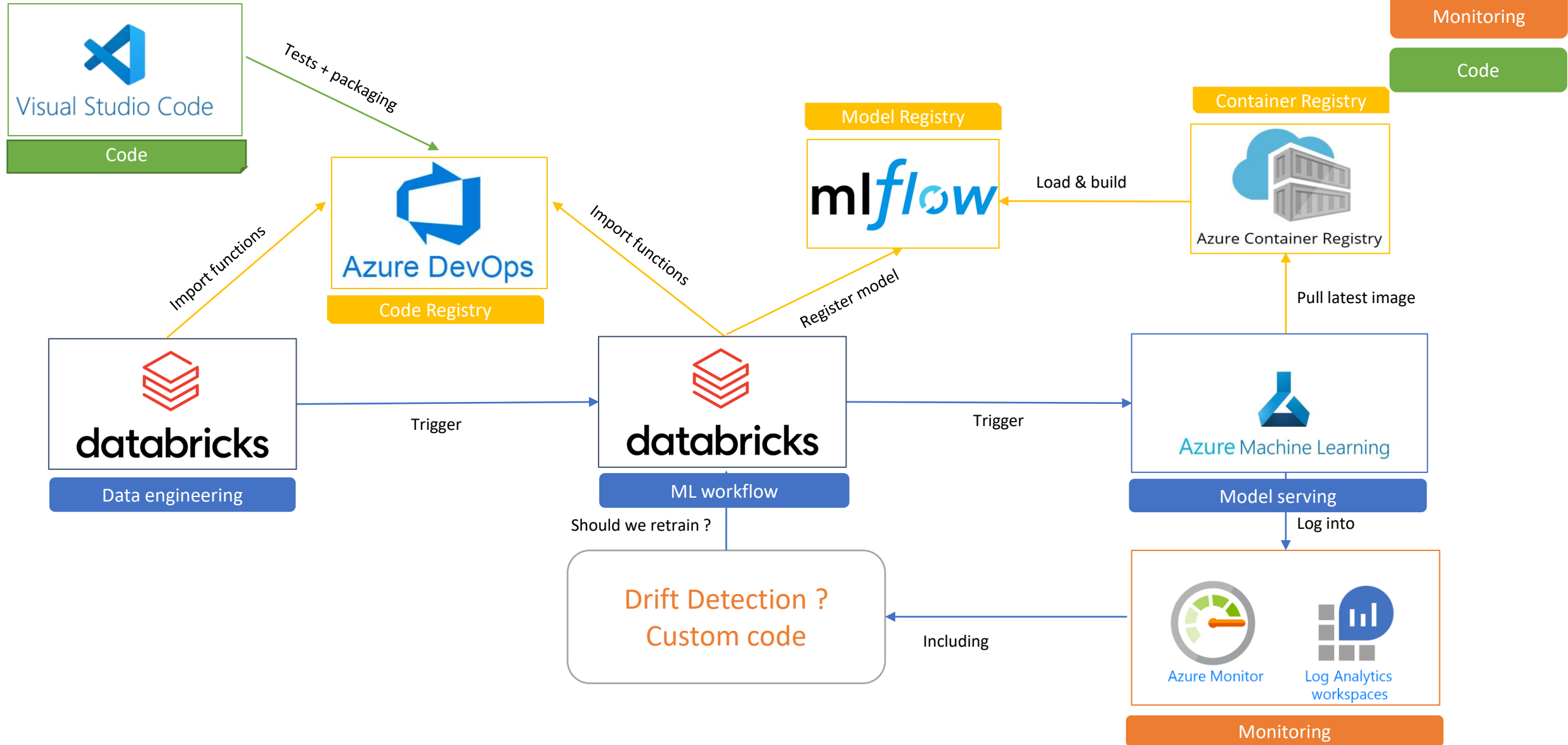




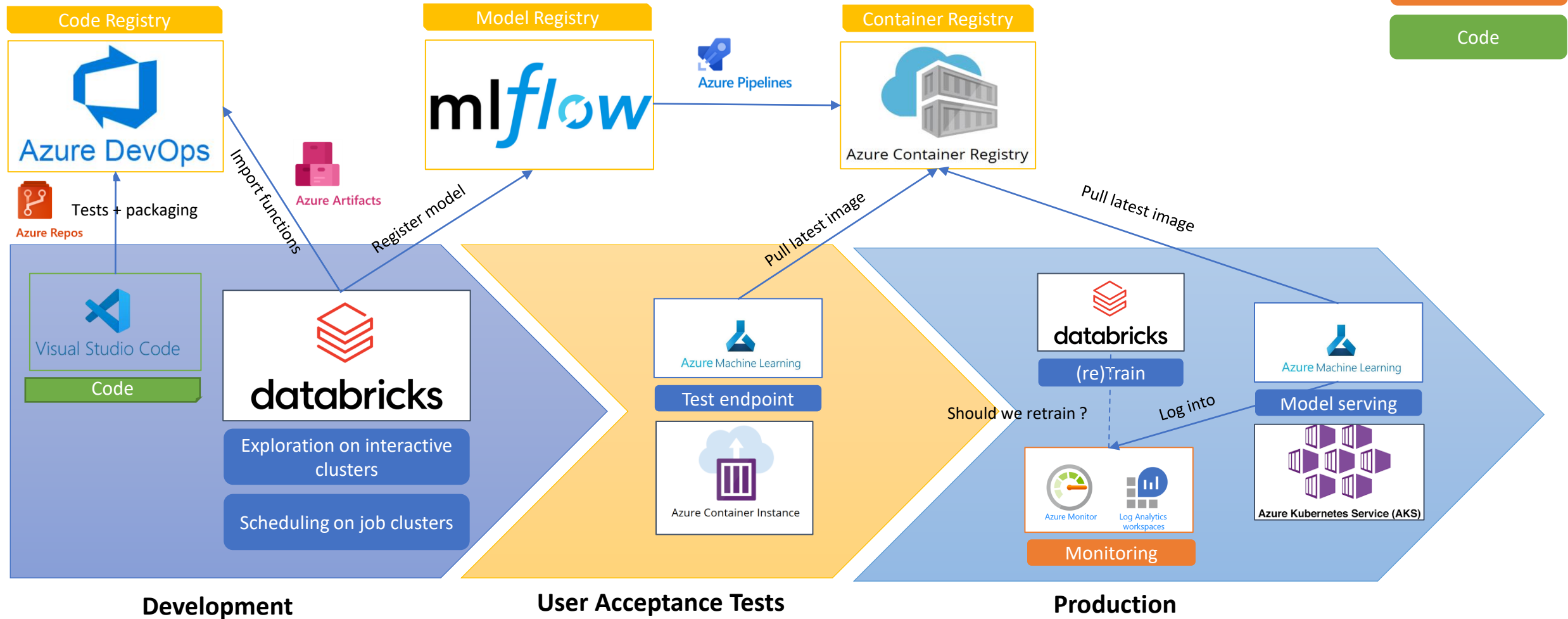
# Our version of MLOps Stack template



# Our MLOps Stack on Microsoft Azure

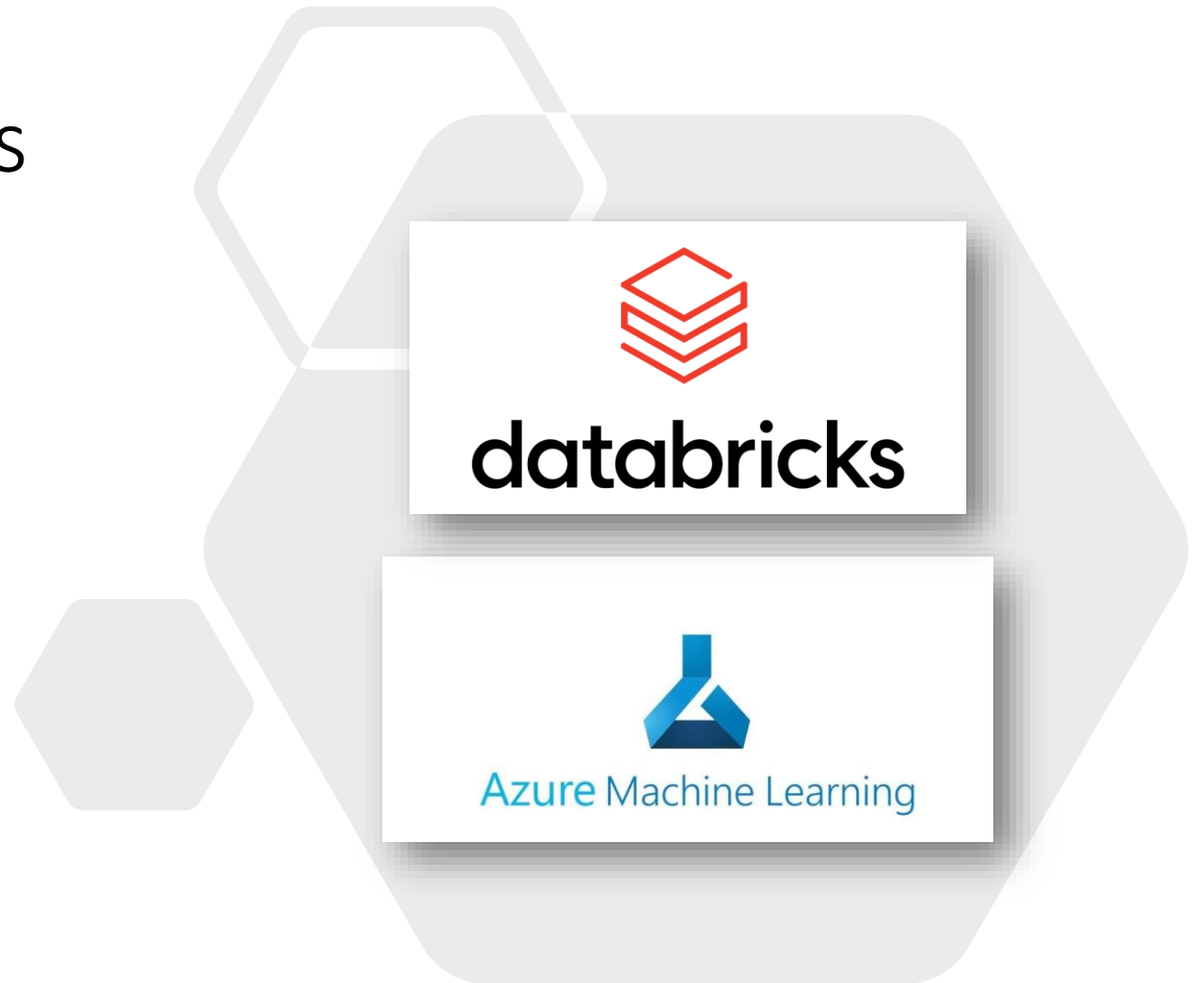


# In a multiple environments context



# Azure ML + Databricks

1. Authentication
2. MLFlow tracking URI
3. Attached compute



# How to authenticate with a SPN



databricks



Azure Active Directory

```
from azureml.core.authentication import ServicePrincipalAuthentication

tenant_id = '8[REDACTED]7'
service_principal_id = '[REDACTED]9'

subscription_id = '[REDACTED]'
resource_group = 'rg-centralized-registry'
workspace_name = 'centralizedaml'

svc_pr = ServicePrincipalAuthentication(
    tenant_id=tenant_id,
    service_principal_id=service_principal_id,
    service_principal_password=svc_pr_password)

ws = Workspace(
    subscription_id=subscription_id,
    resource_group=resource_group,
    workspace_name=workspace_name,
    auth=svc_pr
)

print("Found workspace {} at location {}".format(ws.name, ws.location))
```

# MLFlow : specify the tracking URI

Microsoft Azure | Databricks Portal paul.peton@live.fr

## create model on centralized registry Python

Detached | File | Edit | View: Standard | Run All | Clear

Comments Experiment Revision history

```

1 dbutils.secrets.list('prd-registry-scope')

Out[14]: [SecretMetadata(key='prd-host'),
SecretMetadata(key='prd-token'),
SecretMetadata(key='prd-workspace-id')]

Command took 0.53 seconds -- by paul.peton@live.fr at 24/01/2022, 21:37:00 on unknown cluster

Cmd 17
1 scope = 'prd-registry-scope'
2 key = 'prd'
3
4 registry_uri = 'databricks://' + scope + ':' + key if scope and key else None
5 print(registry_uri)

databricks://prd-registry-scope:prd

Command took 0.02 seconds -- by paul.peton@live.fr at 21/01/2022, 21:52:39 on unknown cluster

Cmd 18
1 mlflow.set_registry_uri(registry_uri)
2 mlflow.register_model(model_uri=f'runs://{run_id}/{artifact_path}', name=model_name)

Successfully registered model 'dev_sample_model'.
=== Copying model files from the source location to the model registry workspace. ===
=== Source model files were copied to dbfs:/databricks/mlflow/tmp-external-source/ba8fe202188046cf99e5aeeclb10a616/dev_sample_artifact in the model registry workspace. You may want to delete the files once the model version is in 'READY' status. You can also find this location in the 'source' field of the created model version. ===
2022/01/21 21:04:59 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation.
Created version '1' of model 'dev_sample_model'.
Out[14]: <ModelVersion: creation_timestamp=1642799099223, current_stage=None, description='', last_updated_timestamp=1642799099916, name='dev_sample_model', run_id='ba8fe202188046cf99e5aeeclb10a616', run_link='https://adb-2912797461624344.4.azuredatabricks.net/?o=2912797461624344#mlflow/experiments/4275780738888038/runs/ba8fe202188046cf99e5aeeclb10a616', source='dbfs:/databricks/mlflow/tmp-external-source/ba8fe202188046cf99e5aeeclb10a616/dev_sample_artifact', status='READY', status_message='', tags={}, user_id='paul.peton@live.fr', version='1'>

Command took 10.65 seconds -- by paul.peton@live.fr at 21/01/2022, 22:04:55 on unknown cluster

Cmd 19
1 # Instantiate an MlflowClient pointing to the local tracking server
2 from mlflow.tracking.client import MlflowClient
3
4 client = MlflowClient(tracking_uri=None, registry_uri=registry_uri)
  
```

# Databricks : link Azure ML workspace



databricks



Azure Machine Learning



Launch Workspace

Documentation



Getting Started



Import Data from File



Import Data from Azure Storage



Notebook



Admin Guide



[Link Azure ML workspace](#)



## ^ Essentials

Status : Active

Resource group : [rg-dbx-dev](#)

Location : East US 2

Subscription : [Microsoft Azure Sponsorship](#)

Subscription ID : [REDACTED]

Tags ([edit](#)) : env : dev MLWorkspaceLinkUpdateTime : 05/11/2022 12:08:14 +00:00

Managed Resource Group : [databricks-rg-dbxdev-mjz7yr6n4vi2q](#)

URL : <https://adb-2912797461624344.4.azuredatabricks.net>

Pricing Tier : standard

Azure ML workspace : [centralizedaml](#)

centralizedaml



databricks



Azure Machine Learning

# Use the tracking URI of Azure ML

## Tracking URI : Azure ML

Cmd 76

```
azureml_tracking_uri = ws.get_mlflow_tracking_uri()
print(f"Azure ML Tracking URI: {azureml_tracking_uri}")

mlflow.set_tracking_uri(azureml_tracking_uri)
print("MLflow Tracking URI:", mlflow.get_tracking_uri())

client = mlflow.tracking.MlflowClient()
```

```
Azure ML Tracking URI: azureml://eastus2.api.azureml.ms/mlflow/v1.0/subscriptions/f80606e5-788f-4dc3-a9ea-2eb9a7836082/resourceGroups/rg-centralized-reg
istry/providers/Microsoft.MachineLearningServices/workspaces/centralizedaml?
MLflow Tracking URI: azureml://eastus2.api.azureml.ms/mlflow/v1.0/subscriptions/f80606e5-788f-4dc3-a9ea-2eb9a7836082/resourceGroups/rg-centralized-regis
try/providers/Microsoft.MachineLearningServices/workspaces/centralizedaml?
```

Command took 0.10 seconds -- by paul.peton@live.fr at 31/05/2022, 13:35:40 on cluster-dev

Cmd 77

```
experiment_name = 'Yellow_Taxi_fromDBX_experiment'
mlflow.set_experiment(experiment_name)
```

```
2022/05/31 11:36:27 INFO mlflow.tracking.fluent: Experiment with name 'Yellow_Taxi_fromDBX_experiment' does not exist. Creating a new experiment.
Out[182]: <Experiment: artifact_location='', experiment_id='d140cd27-08d8-4638-ab62-bf60ca444378', lifecycle_stage='active', name='Yellow_Taxi_fromDBX_e
xperiment', tags={}>
```

Command took 1.04 seconds -- by paul.peton@live.fr at 31/05/2022, 13:36:27 on cluster-dev



# Attached compute for Azure ML



## Attach your compute target and manage your platform using Azure Machine Learning

Bring your own compute like an HDInsight cluster, a Virtual Machine, or a Databricks cluster to use as a compute target with your Azure Machine Learning workspace. [Learn more](#)

+ New

- Azure Databricks
- Data Lake Analytics
- HDInsight
- Kubernetes (preview)
- Synapse Spark pool (preview)
- Virtual machine

### Attach Databricks compute

Compute name \*

Subscription \*

Microsoft Azure Sponsorship

[Refresh subscriptions](#)

Databricks workspace \*

Search or select an Azure Databricks workspace

[Refresh Databricks workspaces](#)

Databricks access token \*

# Launch a Pipeline step on Databricks

```
Répertoire par défaut > centralizedaml > Compute >

from azureml.core.compute import DatabricksCompute
from azureml.pipeline.steps import DatabricksStep

databricks_compute = DatabricksCompute(workspace=ws, name="databricks-prod")
db_cluster_id = "0530-190209-eievpfbj"
notebook_path = os.getenv("DATABRICKS_NOTEBOOK_PATH", "/Shared/hello")

dbNbStep = DatabricksStep(
    name="DBXNotebookInWS",
    existing_cluster_id=db_cluster_id,
    num_workers=4,
    notebook_path=notebook_path,
    run_name='DBX_notebook_run',
    compute_target=databricks_compute,
    allow_reuse=False
)
```

```
steps = [dbNbStep]
```

```
from azureml.pipeline.core import Pipeline

pipeline = Pipeline(workspace = ws, steps = [steps])
print('Pipeline is built')
pipeline.validate()
print('Pipeline validation complete')
pipeline_run = exp.submit(pipeline)
print('Pipeline is submitted for execution')

pipeline_run.wait_for_completion(show_output = False)
```



**Azure Synapse Analytics**

And what about  
Synapse Analytics +  
Azure ML ?



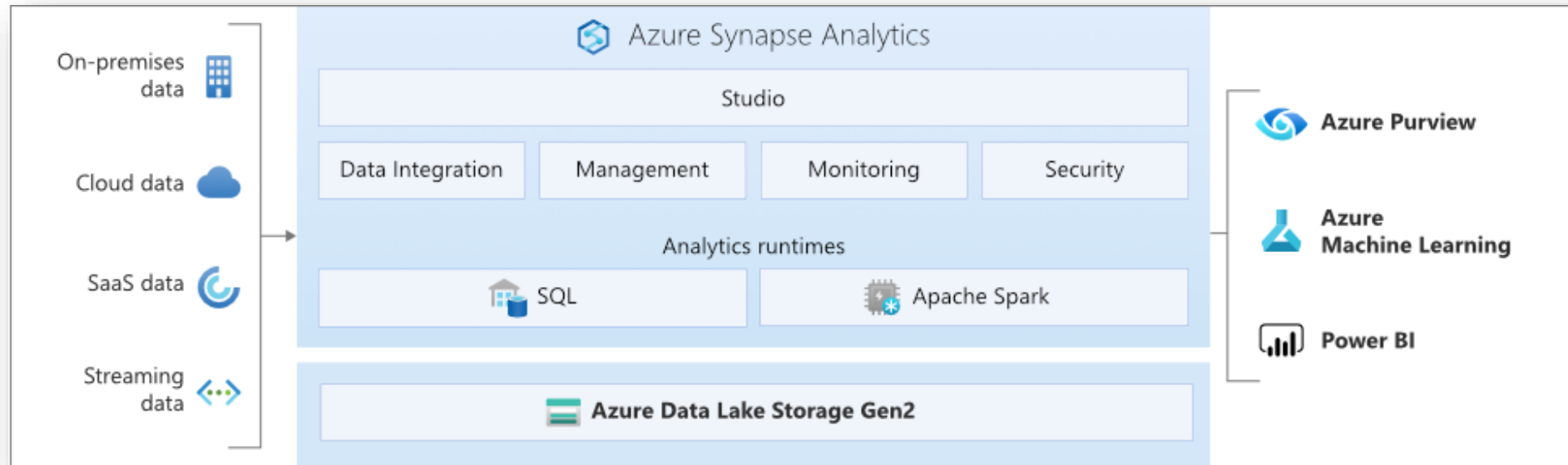
**Azure Machine Learning**

# Azure Synapse Analytics in a few words



Azure Synapse Analytics

The data toolbox **Unified experience** for data project on Azure



Azure Synapse brings together the best of **SQL** technologies used in data warehousing :

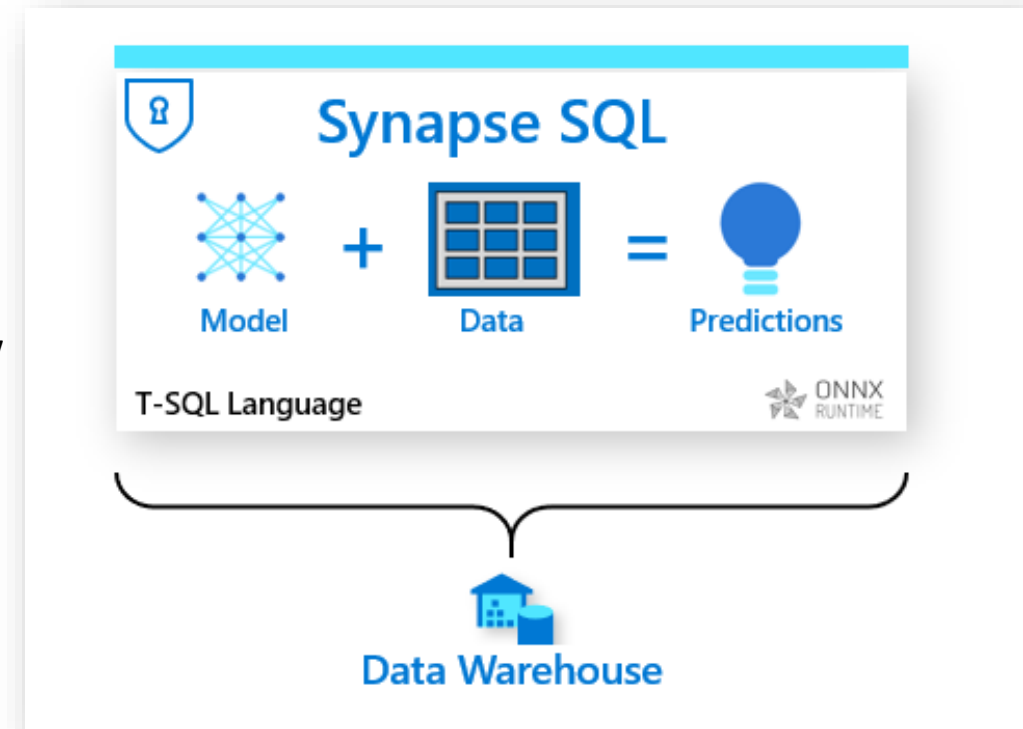
**Spark** technologies used for big data

**Pipelines** for data integration and ETL/ELT

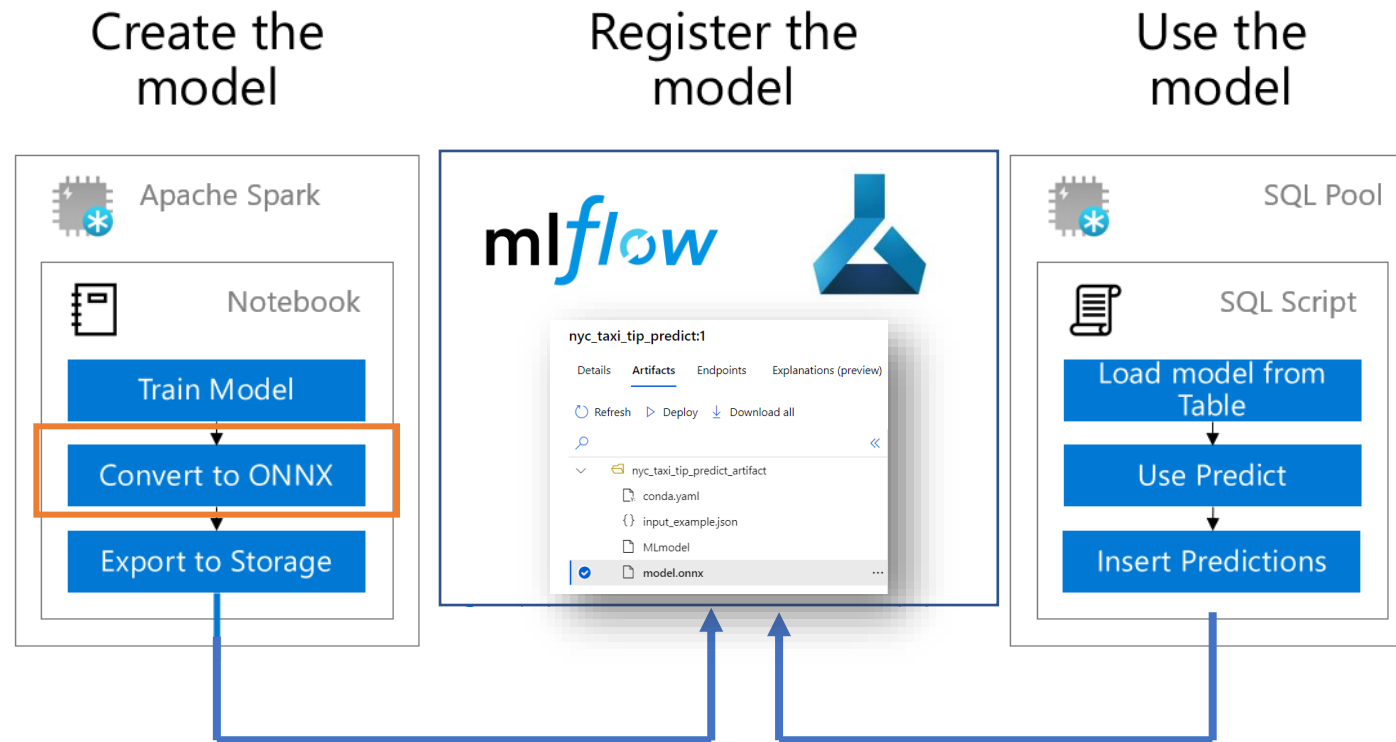
deep integration with other Azure services such as **Power BI**, **CosmosDB**, and **AzureML**.

# Use ONNX model in Synapse

- Dedicated SQL pool expects a **pre-trained** model
  - only supports **ONNX** format models
  - The scoring data needs to be in the **same format** as the training data
  - Make sure that the **names** and **data types** of the model inputs match the column names and data types of the new prediction data.
- The model is stored in a dedicated SQL pool user table as a **hexadecimal string**.
- Use the **T-SQL PREDICT** function to score the model



# Machine Learning workflow in Synapse Analytics



# Linked services : from Synapse to Azure ML

The screenshot displays the Microsoft Azure Synapse Analytics interface. The top navigation bar shows 'Microsoft Azure | Synapse Analytics | satraining'. The left sidebar contains various navigation options: Analytics pools, SQL pools, Apache Spark pools, External connections, Linked services (selected), Azure Purview (Preview), Integration, Triggers, Integration runtimes, Security, Access control, Credentials, Managed private endpoints, Code libraries, Workspace packages, Source control, and Git configuration.

The main content area is titled 'Linked services'. It includes a '+ New' button and a search bar labeled 'Filter by name'. Below the search bar, it says 'Showing 1 - 3 of 3 items'. A table lists the linked services:

Name	Type	Related
HTTPnycitibike	HTTP	1
satraining-WorkspaceDefaultSqlServer	Azure Synapse Analytics	0
satraining-WorkspaceDefaultStorage	Azure Data Lake Storage Gen2	8

Below the table, there is a 'New linked service' dialog box. It has a search bar with 'machine' entered and tabs for 'All', 'Azure', 'Compute', 'Database', 'File', 'Generic protocol', and 'NoSQL'. The 'Compute' tab is selected, showing two options: 'Azure Machine Learning' (selected) and 'Azure Machine Learning'.

On the right, a larger 'New linked service (Azure Machine Learning)' dialog box is open. It contains the following fields and options:

- Name \***: AzureMLServiceTraining
- Description**: (empty text area)
- Connect via integration runtime \***: AutoResolveIntegrationRuntime
- Azure Machine Learning workspace selection method**: ☒ From Azure subscription, ☐ Enter manually
- Azure subscription**: Microsoft Azure Sponsorship (f80606e5-788f-4dc3-a9ea-2eb9a7836082)
- Azure Machine Learning workspace name \***: mlworkspace-training
- Tenant \***: 8d...
- Service principal ID \***: (empty text field)
- Service principal key**: Azure Key Vault
- Buttons**: Commit, Back, Test connection, Cancel

# SQL Pool: enrich table with ML model

The screenshot displays the Microsoft Azure Synapse Analytics interface. The top navigation bar shows 'Microsoft Azure | Synapse Analytics | satraining' with a search bar and user profile 'paul.peton@live.fr'. The left sidebar contains navigation icons for Home, Data, Workspace, and Linked. The main pane shows the 'Data' section with a 'Workspace' tab. Under 'Databases', 'DW100cpool (SQL)' is expanded, showing 'Tables' and 'dbo.Models'. The 'Columns' section for 'dbo.nyc\_taxi' is visible, listing attributes like 'tipped (int, null)', 'fareAmount (float, ...)', 'paymentType (int, ...)', 'passengerCount (int, ...)', 'tripDistance (float, ...)', 'tripTimeSecs (bigint, n...)', and 'pickupTimeBin (nvarchar...)'. A context menu is open over the 'Columns' section, with options: 'New SQL script', 'New notebook', 'New data flow', 'New integration dataset', 'Machine Learning', and 'Refresh'. The 'Machine Learning' option is selected, and a tooltip 'Enrich with existing model' is shown. On the right, a modal window titled 'Enrich with existing model' for 'dbo.nyc\_taxi' is displayed. It prompts to 'Select the model you want to use to enrich the selected dataset.' and shows 'Azure Machine Learning' workspace 'mlworkspace-training'. A table lists available models:

Name	Versi...	Created	Created by	Fra...
nyc_taxi_tip_predict	1	05:40:46 05/1...	Paul PETON	Cust...



# SQL Pool: Enrich table with existing model

### Enrich with existing model

dbo.nyc\_taxi

Map the source table columns to the expected model inputs. [Learn more](#)

#### Input mapping \*

+ New | Delete

<input type="checkbox"/>	Source column		Model input	Input type		
<input type="checkbox"/>	fareAmount	→	fareAmount	real	+	🗑️
<input type="checkbox"/>	paymentType	→	paymentType	bigint	+	🗑️
<input type="checkbox"/>	passengerCount	→	passengerCount	bigint	+	🗑️
<input type="checkbox"/>	tripDistance	→	tripDistance	real	+	🗑️
<input type="checkbox"/>	tripTimeSecs	→	tripTimeSecs	bigint	+	🗑️
<input type="checkbox"/>	pickupTimeBin	→	pickupTimeBin	varchar	+	🗑️

#### Output mapping \*

+ New | Delete

<input type="checkbox"/>	Model output	Output type	
<input type="checkbox"/>	output_label	bigint	+ 🗑️

Continue

Back

Cancel

### Enrich with existing model

dbo.nyc\_taxi

#### Stored procedure

A stored procedure will be created once you run the generated script. Specify a name for this stored procedure. [Learn more](#)

Stored procedure name \*

sp\_taxi\_tip\_model

#### Target table

Create a new database table or use an existing table to store the machine learning model. [Learn more](#)

#### Select target table \*

☐ Existing table ☒ Create new

New table \*

model\_storage

Deploy model + open script

Back

Cancel

# Script: stored procedure

The screenshot displays the Microsoft Azure Synapse Analytics interface. The top navigation bar includes the Microsoft Azure logo, the path 'Synapse Analytics > satraining', a search bar, and user information 'paul.peton@live.fr' with the text 'RÉPERTOIRE PAR DÉFAUT'. Below the navigation bar, the left sidebar shows the 'Data' section with a 'Workspace' tab and a 'Linked' tab. The 'Databases' section is expanded, showing 'DW100cpool (SQL)' with a list of tables: 'dbo.Models', 'dbo.nyc\_taxi', 'External tables', 'External resources', 'Views', 'Programmability', 'Schemas', 'Security', and 'default (Spark)'. The main area displays a SQL script titled 'SQL script 1'. The script is as follows:

```
1 CREATE PROCEDURE sp_taxi_tip_model
2 AS
3 BEGIN
4
5 SELECT
6     CAST([fareAmount] AS [real]) AS [fareAmount],
7     CAST([paymentType] AS [bigint]) AS [paymentType],
8     CAST([passengerCount] AS [bigint]) AS [passengerCount],
9     CAST([tripDistance] AS [real]) AS [tripDistance],
10    [tripTimeSecs],
11    CAST([pickupTimeBin] AS [varchar]) AS [pickupTimeBin]
12 INTO [dbo].[#nyc_taxi]
13 FROM [dbo].[nyc_taxi];
14
15 SELECT *
16 FROM PREDICT (MODEL = (SELECT [model] FROM model_storage WHERE [ID] = 'nyc_taxi_tip_predict:1'),
17              DATA = [dbo].[#nyc_taxi],
18              RUNTIME = ONNX) WITH ([output_label] [bigint])
19
20 END
21 GO
22
23 EXEC sp_taxi_tip_model
```

The script is executed against the 'DW100cpool' database. The 'SELECT \*' statement is highlighted with an orange box. The script is executed against the 'DW100cpool' database.

# Table for model storage

Microsoft Azure | Synapse Analytics | satraining

Search

master branch | Validate all | Commit all | Publish

## Data

Workspace Linked

Filter resources by name

- Databases 2
  - DW100cpool (SQL)
    - Tables
      - dbo.model\_storage
        - Columns
          - ID (nvarchar(1024), not null)
          - name (nvarchar(1024), not null)
          - description (nvarchar(1024), null)
          - version (int, null)
          - created\_time (datetime2(7), null)
          - created\_by (nvarchar(128), null)
          - framework (nvarchar(64), null)
          - model (varbinary(max), null)**
          - inputs\_schema (nvarchar(max), null)
          - outputs\_schema (nvarchar(max), null)

CREATE PROCEDURE ... SQL script 1

Run | Undo | Commit | Query plan | Connect to DW100cpool | Use database DW100cpool

```
1 SELECT TOP (100) [ID]
2 ,[name]
3 ,[description]
4 ,[version]
5 ,[created_time]
6 ,[created_by]
7 ,[framework]
8 ,[model]
9 ,[inputs_schema]
10 ,[outputs_schema]
11 FROM [dbo].[model_storage]
```

### Results

Messages

View Table Chart | Export results

Search

ID	name	description	version	created_time	created_by	framework	model	inputs_schen
nyc_taxi_tip_pre...	nyc_taxi_tip_pre...	NULL	1	2021-05-18T05:...	aa7d2357-2b1f...	Custom	0x08071208736...	[{"name": "fa

00:00:01 Query executed successfully.

# T-SQL PREDICT function

Microsoft Azure | Synapse Analytics | satraining

Search

master branch | Validate all | Commit all | Publish

Data

Workspace | Linked

Filter resources by name

Databases 2

DW100cpool (SQL)

Tables

- dbo.model\_storage
- dbo.Models
- dbo.nyc\_taxi
- dbo.nyc\_taxi\_types

External tables

External resources

Views

Programmability

Schemas

Security

default (Spark)

SQL query for ML pr...

Run | Undo | Committed | Query plan | Connect to DW100cpool | Use database DW100cpool

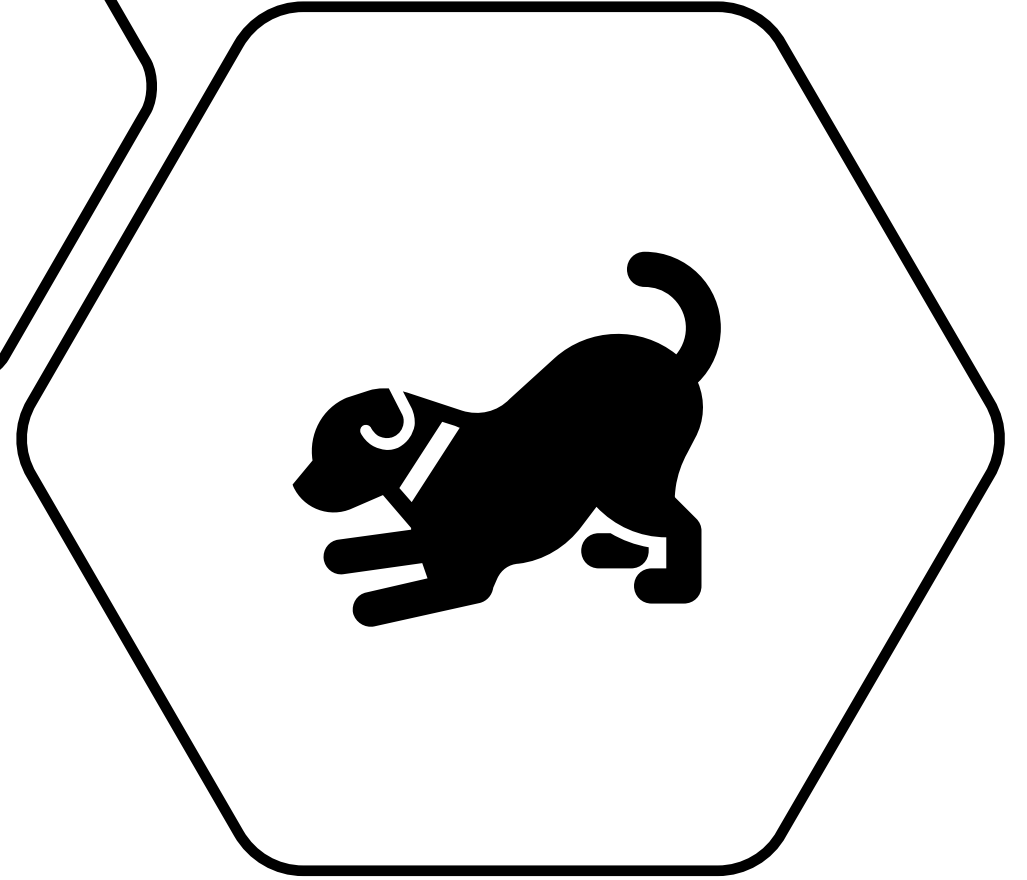
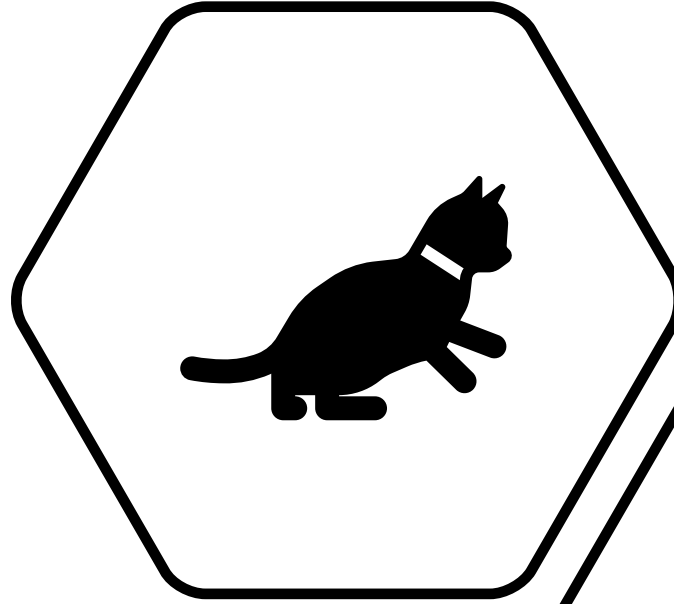
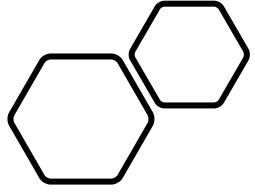
```
1 -- Query for ML predictions
2 SELECT
3 d.[tipped]
4 ,d.[fareAmount]
5 ,d.[paymentType]
6 ,d.[passengerCount]
7 ,d.[tripDistance]
8 ,d.[tripTimeSecs]
9 ,d.[pickupTimeBin]
10 ,p.output_label
11 FROM PREDICT(MODEL = ( SELECT model FROM model_storage WHERE Id = 'nyc_taxi_tip_predict:1'),
12 DATA = [dbo].[nyc_taxi_types] AS d, RUNTIME = ONNX) WITH (output_label bigint) AS p;
13
```

Results | Messages

View | Table | Chart | Export results

tipped	fareAmount	paymentType	passengerCount	tripDistance	tripTimeSecs	pickupTimeBin	output_label
1	5	1	3	0.7	240	Afternoon	1
0	4.5	2	1	0.9	211	PMRush	0
1	31	1	1	9.5	1973	Night	1
0	40	2	2	13.2	2059	Night	0

00:00:01 Query executed successfully.



So, friend or foe ?

# The main differences



## Dedicated to Machine Learning

- By and for **Data Scientists**
  - Explore on compute instances
  - Simple batch or real time scenarios
- Guided for the industrialization of their models and pipelines
  - Data versioning (?)
  - Model versioning
  - Web service



## Dedicated to Big Data Science

- Work with big volume
- Version data with Delta format
- Use Spark to benefit of “scale out” power

# The main differences



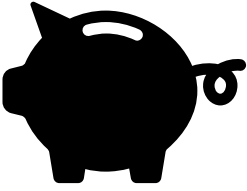
## Dedicated to Machine Learning

- By and for **Data Scientists**
  - Explore on compute instances
  - Simple batch or real time scenarios
- Guided for the industrialization of their models and pipelines
  - Data versioning (?)
  - Model versioning
  - Web service

## Dedicated to Data Engineering

- Data preparation : ETL / ELT
- Orchestration
- Dedicated SQL pool for table storage
- Serverless SQL pool for exploration

# #finops : some tips to save money in Azure



- Azure Machine Learning
  - Schedule stop on compute instances
  - Use compute cluster only for hyperparameter tuning or autoML
- Azure Databricks
  - Use Spot VM
  - Use Databricks Jobs instead of Databricks interactive clusters
- Azure Synapse Analytics
  - Use serverless SQL Pool to explore files with SQL