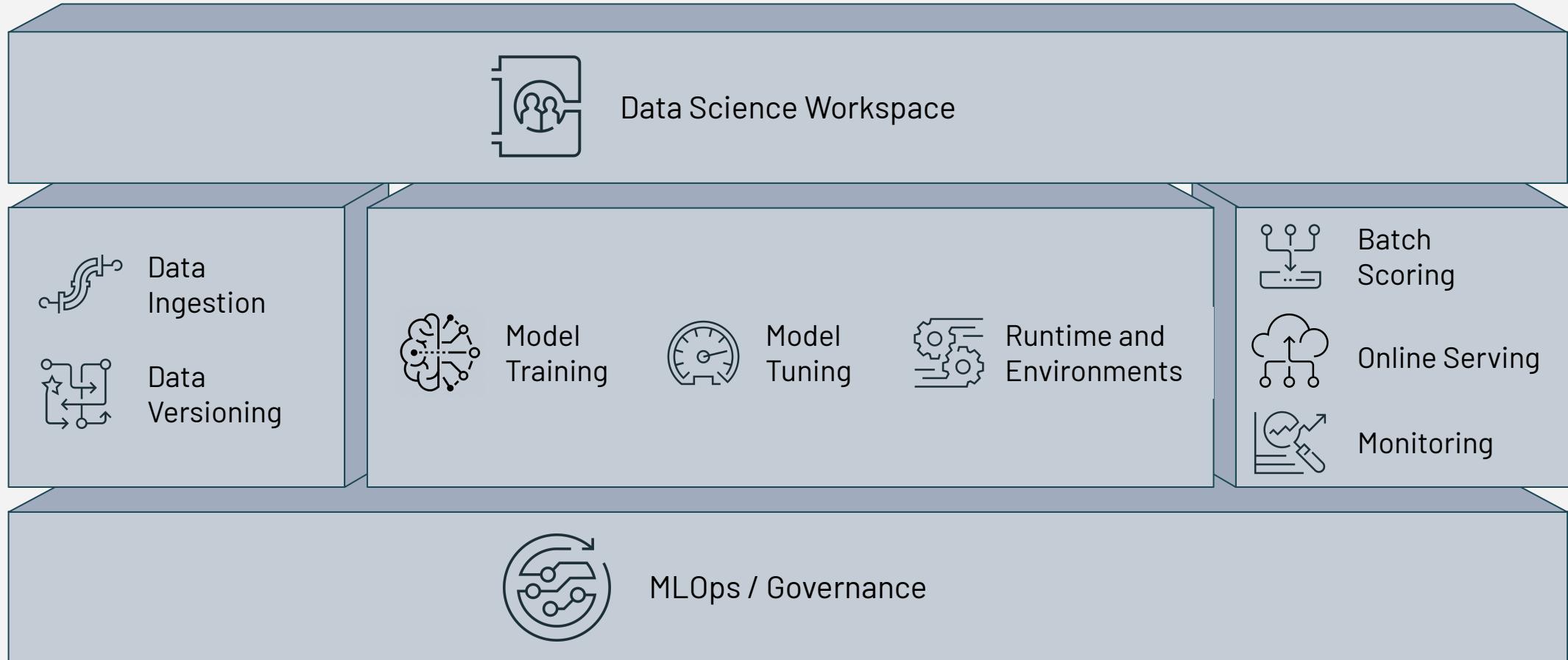


# Databricks Overview for MLOps

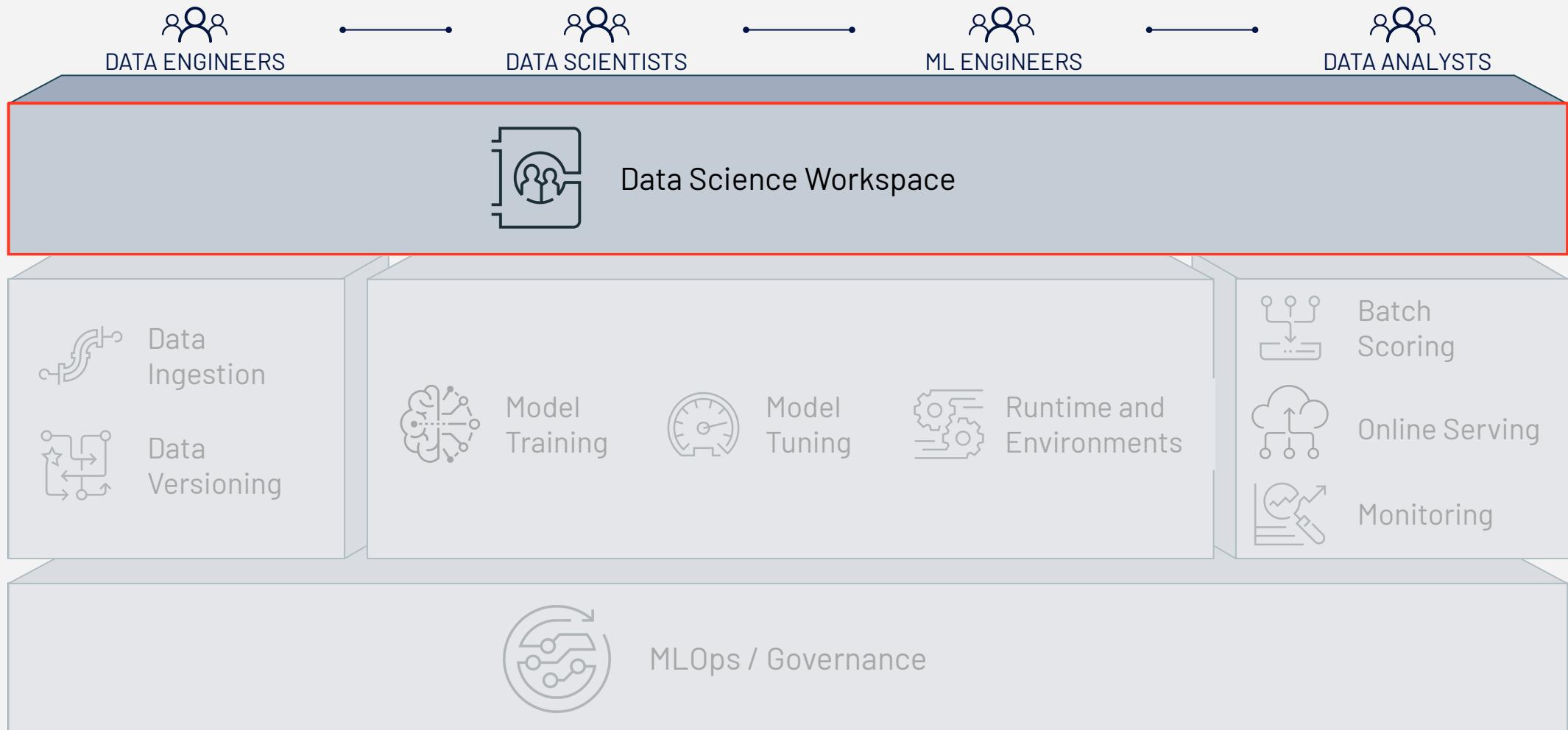


**Clemens Mewald**  
Director of Product Management

# The Databricks ML Platform



# Collaborative Data Science Workspace



# Databricks Notebooks



**Multi-Language**  
Scala, SQL, Python, R: All in one notebook.

A screenshot of the Databricks Notebook interface. The left sidebar shows navigation options like Home, Databricks workspace, Projects, Recents, Data, Clusters, Jobs, Models, and Search. The main area displays a notebook titled "Forecasting Demo (Python)". The code cell contains:

```
13
14 return m, forecast
```

Command took 0.08 seconds -- by clemens.mewald@databricks.com at 11/4/2020, 10:21:04 AM on Clemens ML

Cmd 3

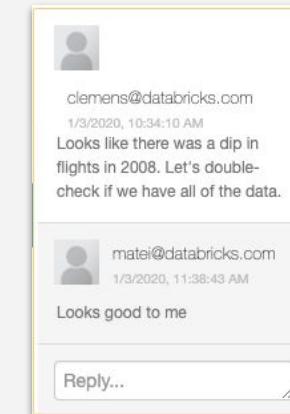
```
1 m, forecast = forecast(df)
```

Below the code is a line plot showing flight count over time from 2008 to 2017. The y-axis ranges from 0 to 350,000, and the x-axis shows years from 2008 to 2017. The plot shows a significant dip around 2008.

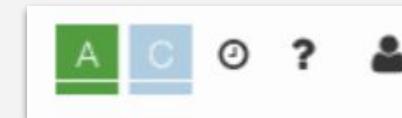
**Collaborative**  
Realtime co-presence, co-editing, and commenting.

## Cloud-native Collaboration Features

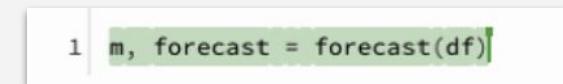
### Commenting



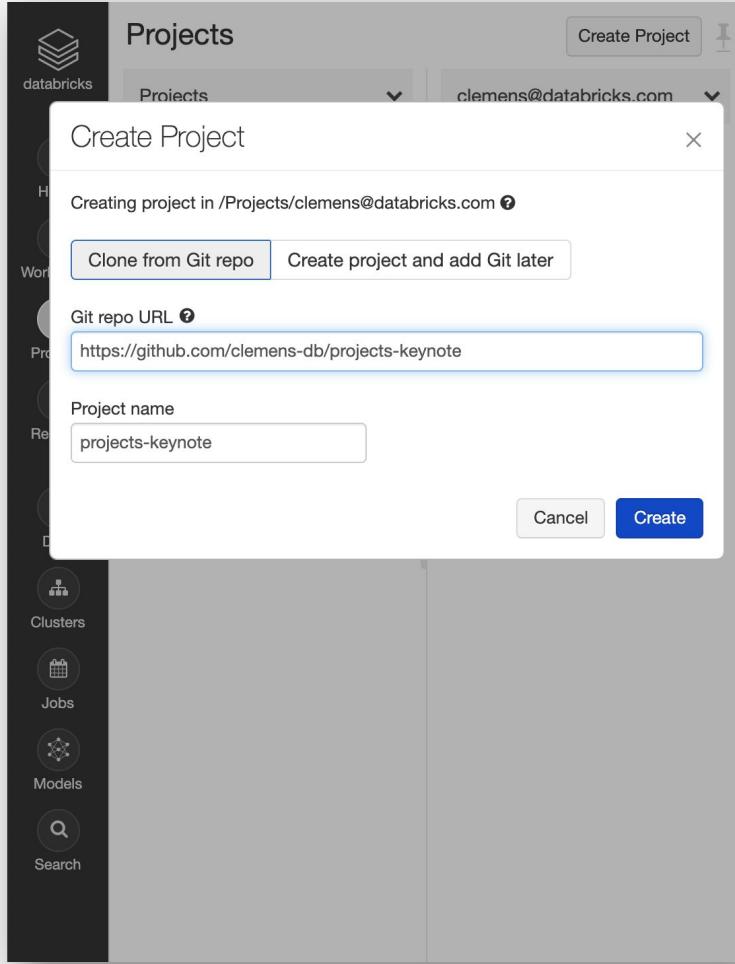
### Co-Presence



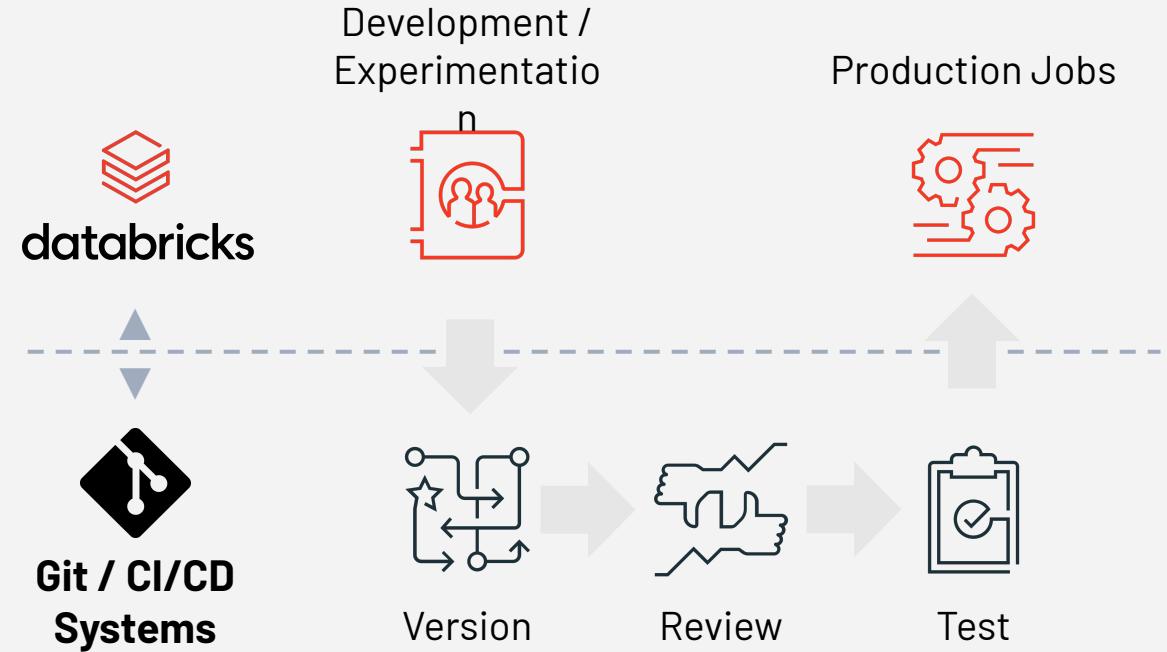
### Co-Editing



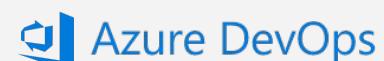
# (Git-based) Projects



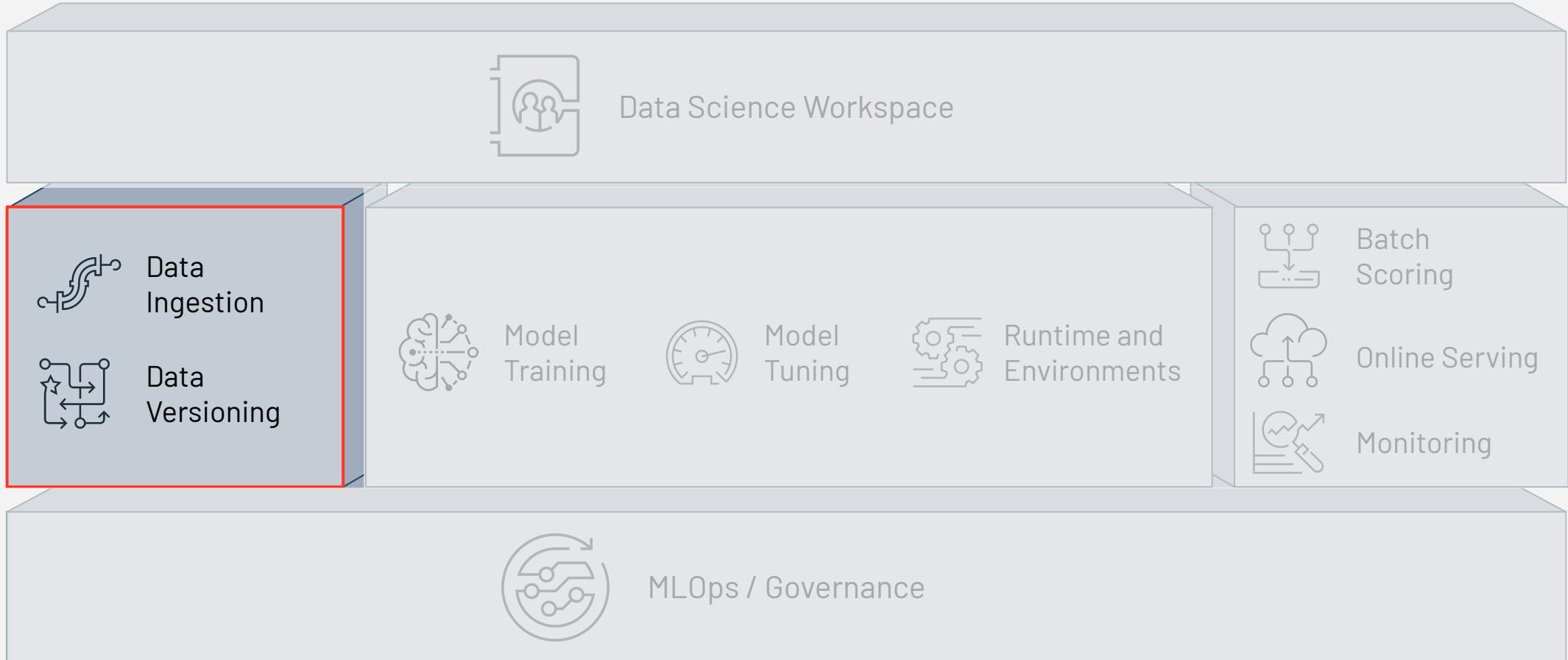
## CI/CD Integration



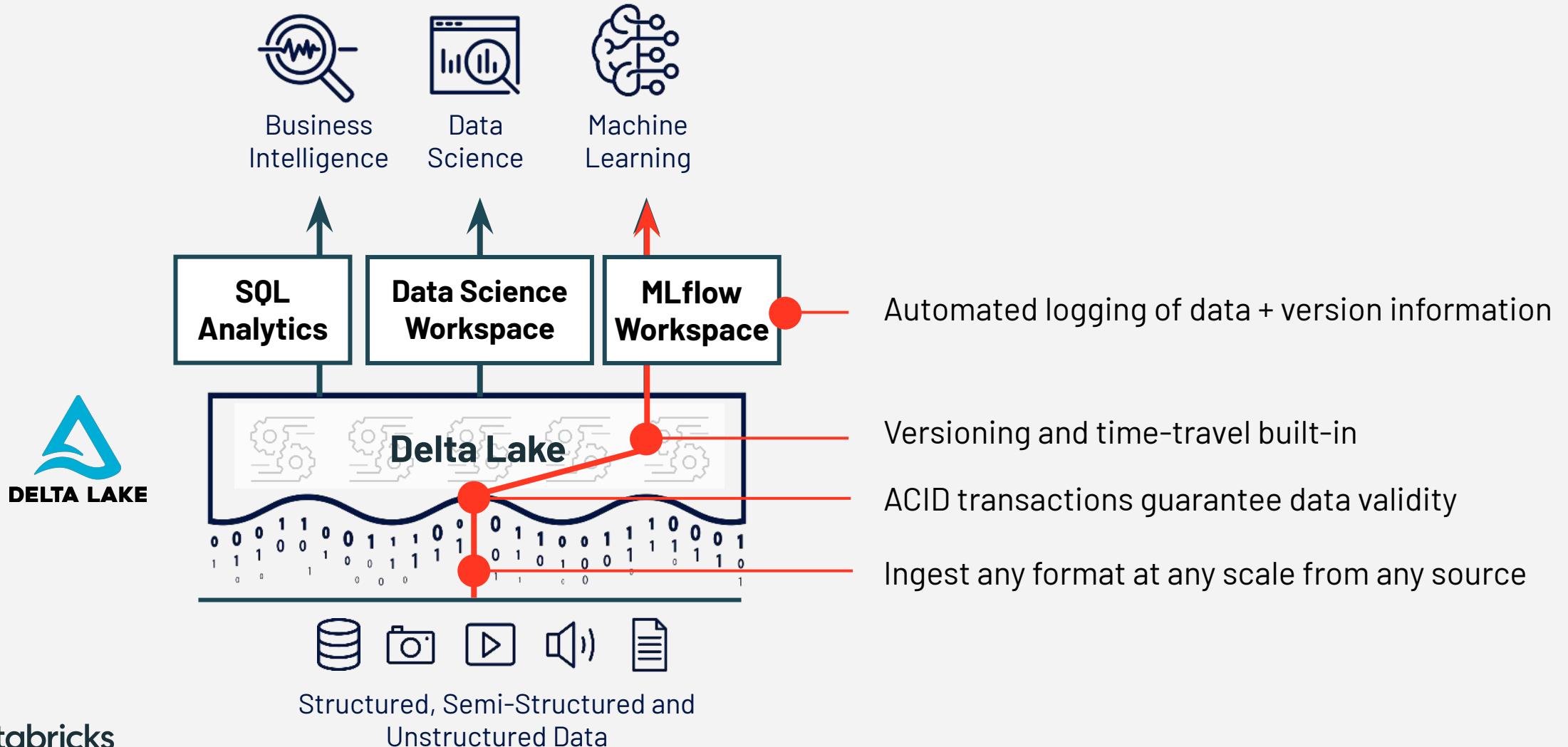
## Supported Git Providers



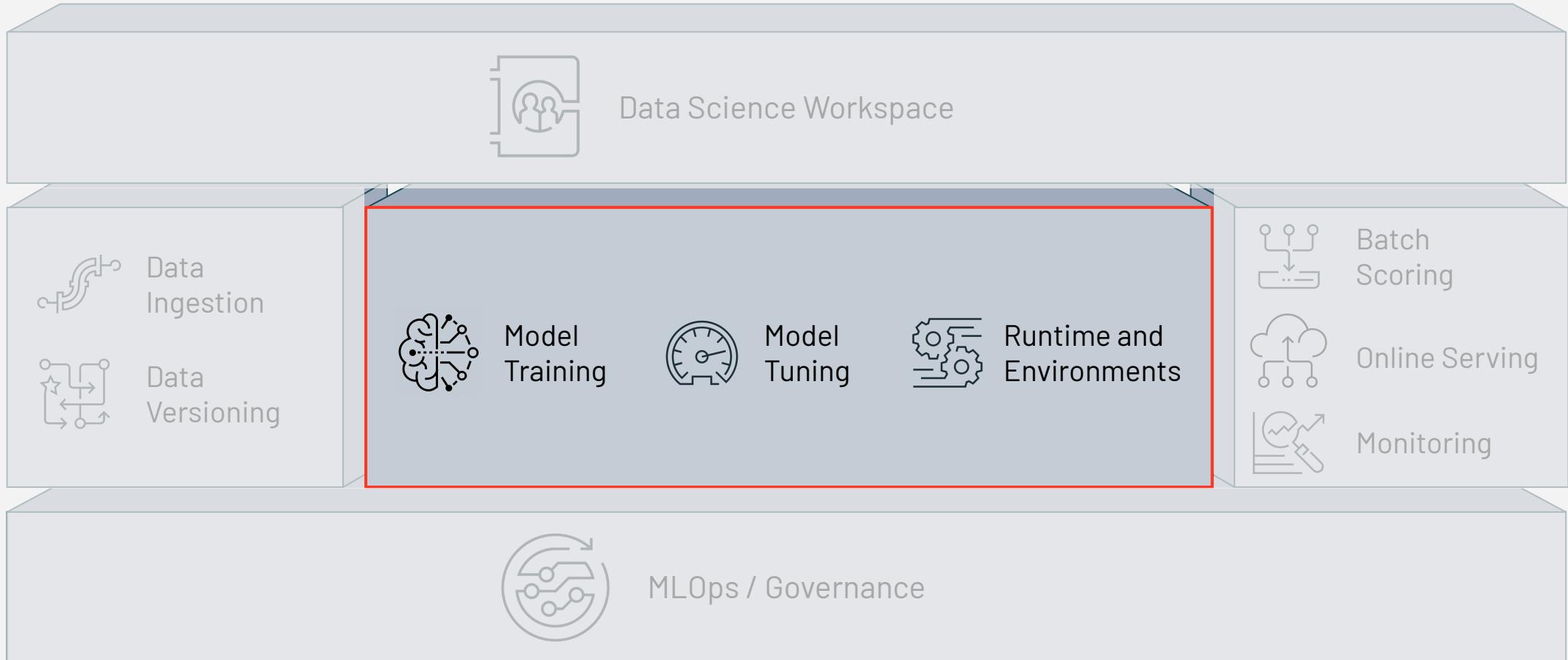
# High Quality Data at Scale



# High Quality Data at Scale



# Turnkey ML Training at Scale



# Turnkey ML Training at Scale

**ML Runtime: DevOps-free Environment  
optimized for Machine Learning**

Databricks Runtime Version [?](#) [Learn more](#)

Runtime: 7.5 ML (GPU, Scala 2.12, Spark 3.0.1) [|](#) NVIDIA EULA [?](#)

Use your own Docker container [?](#)

**Autopilot Options**

Enable autoscaling [?](#)

Enable autoscaling local storage [?](#)

Terminate after  minutes of inactivity [?](#)

**Worker Type** [?](#)

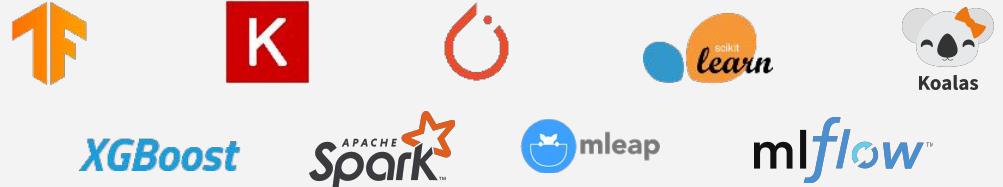
g4dn.xlarge	16.0 GB Memory, 1 GPU, 0.71 DBU	<a href="#"> </a> <a href="#">▼</a>
	Min Workers	Max Workers
	<input type="text" value="2"/>	<input type="text" value="8"/>

**Driver Type**

Same as worker	16.0 GB Memory, 1 GPU, 0.71 DBU	<a href="#"> </a> <a href="#">▼</a>
----------------	---------------------------------	-------------------------------------



**Packages up the most popular ML Toolkits**

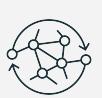


**Simplifies Distributed ML/DL**



Distribute and scale any single-machine ML code to 1,000's of machines.

**Built-in AutoML and Auto-Logging**



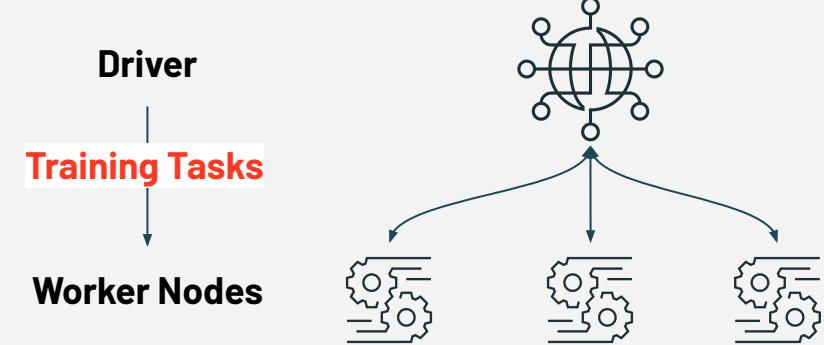
Hyperparameter tuning, AutoML, automated tracking, and visualizations with MLflow

# Distributed Training

- Built-in support in the ML Runtime

## HorovodRunner (Keras, TensorFlow, and PyTorch)

```
hr = HorovodRunner(np=2)  
hr.run(train)
```



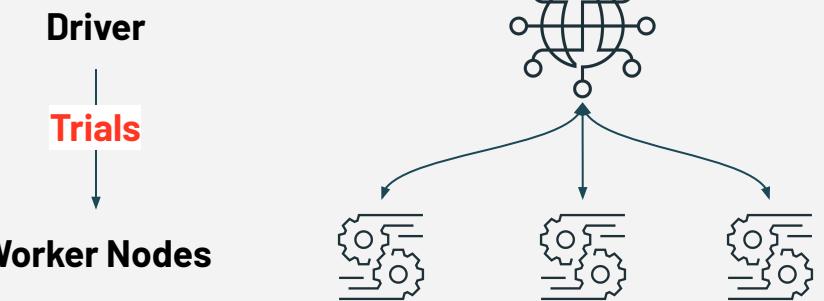
## TensorFlow native Distribution Strategy (Spark TensorFlow Distributor)

```
runner = MirroredStrategyRunner(num_slots=1, local_mode=True, use_gpu=USE_GPU)  
runner.run(train)
```

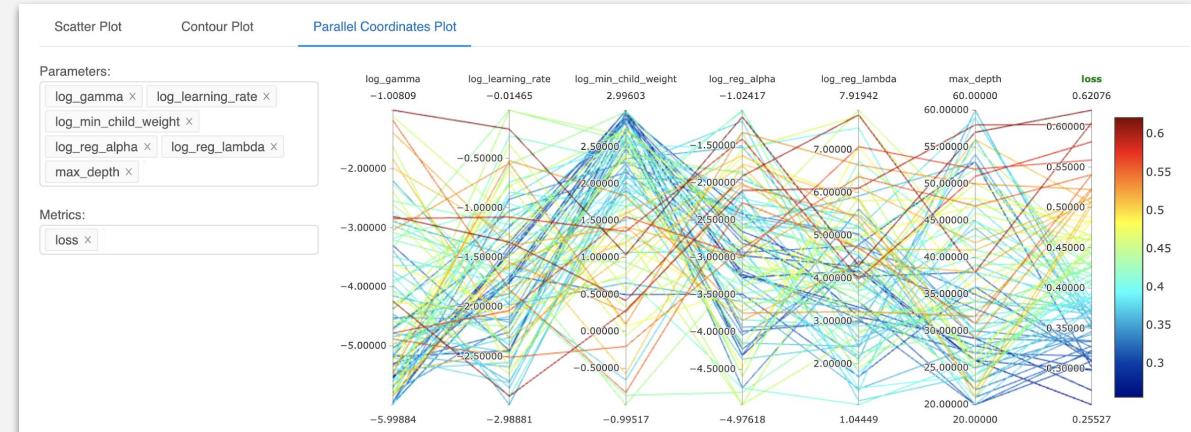
# Distributed Tuning

- Built-in support in the ML Runtime

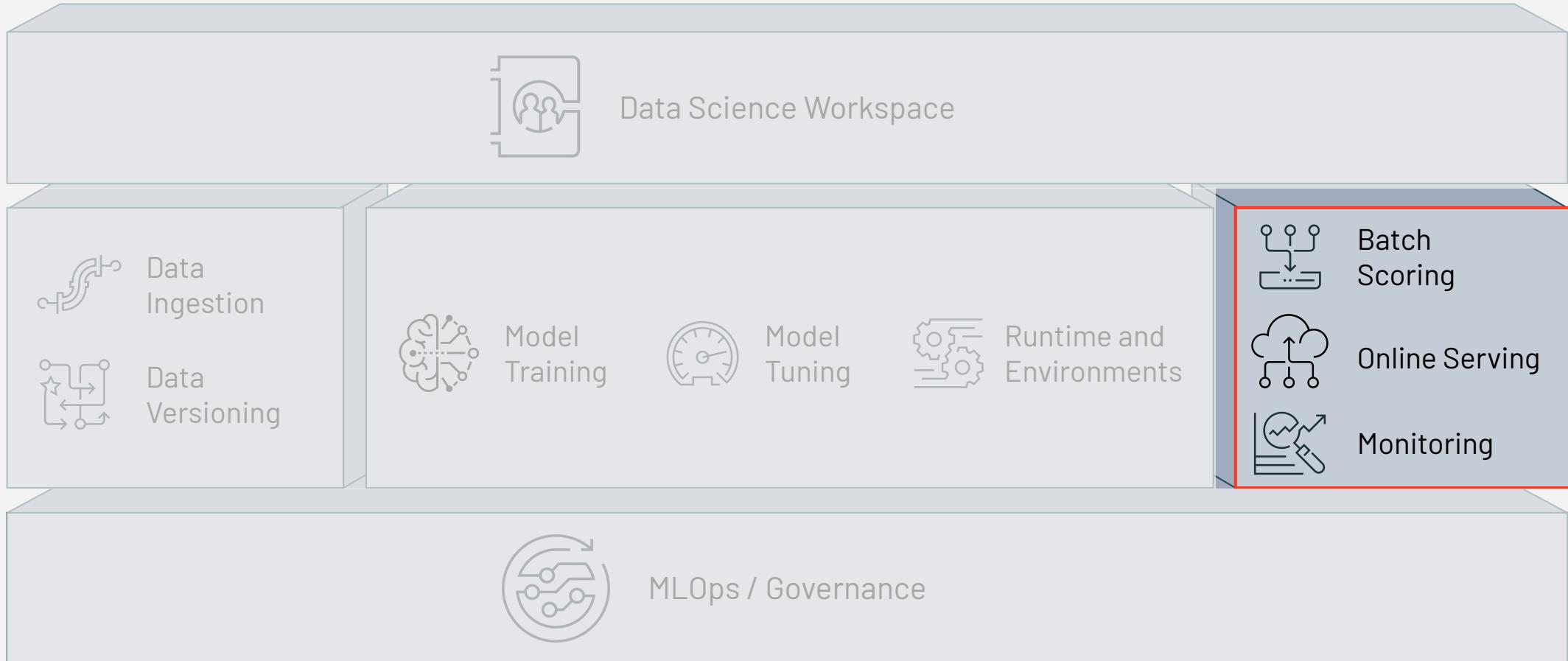
```
search_space = {  
    'max_depth': hp.quniform('max_depth', 20, 60, 1),  
    'log_learning_rate': hp.uniform('log_learning_rate', -3, 0),  
    'log_reg_alpha': hp.uniform('log_reg_alpha', -5, -1),  
    'log_reg_lambda': hp.uniform('log_reg_lambda', 1, 8),  
}  
  
best_params = fmin(fn=train_model,  
                    space=search_space,  
                    algo=tpe.suggest,  
                    max_evals=96,  
                    trials=SparkTrials(parallelism=6))
```



## mlflow™ Integration

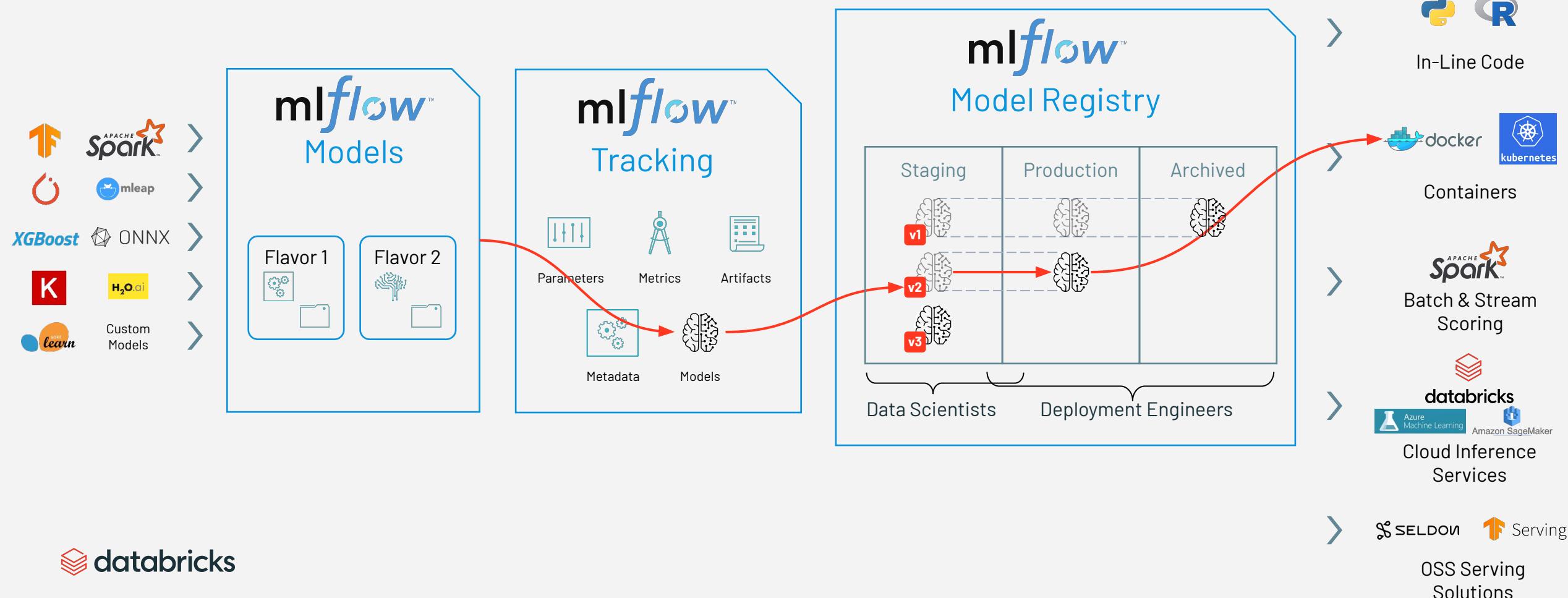


# Support for all Deployment Modes



# Support for all Deployment Modes

**mlflow™**  
Deployment Options



# Support for all Deployment Modes



Deploying an MLLib  
model as a Spark UDF

```
model_udf = mlflow.pyfunc.spark_udf(spark, model_uri='models:/clemens-windfarm-signature/production')
df.with_column('prediction', model_udf)
```

# Support for all Deployment Modes



Deploying an MLLib  
model as a Spark UDF



Deploying a Scikit Learn  
model as a Spark UDF

```
model_udf = mlflow.pyfunc.spark_udf(spark, model_uri='models:/clemens-windfarm-signature/production')
df.with_column('prediction', model_udf)
```

# Support for all Deployment Modes



Deploying an MLlib  
model as a Spark UDF



Deploying a Scikit Learn  
model as a Spark UDF



Deploying a TensorFlow  
model as a Spark UDF

```
model_udf = mlflow.pyfunc.spark_udf(spark, model_uri='models:/clemens-windfarm-signature/production')
df.with_column('prediction', model_udf)
```

# Support for all Deployment Modes



Deploying an MLlib  
model as a Spark UDF



Deploying a Scikit Learn  
model as a Spark UDF

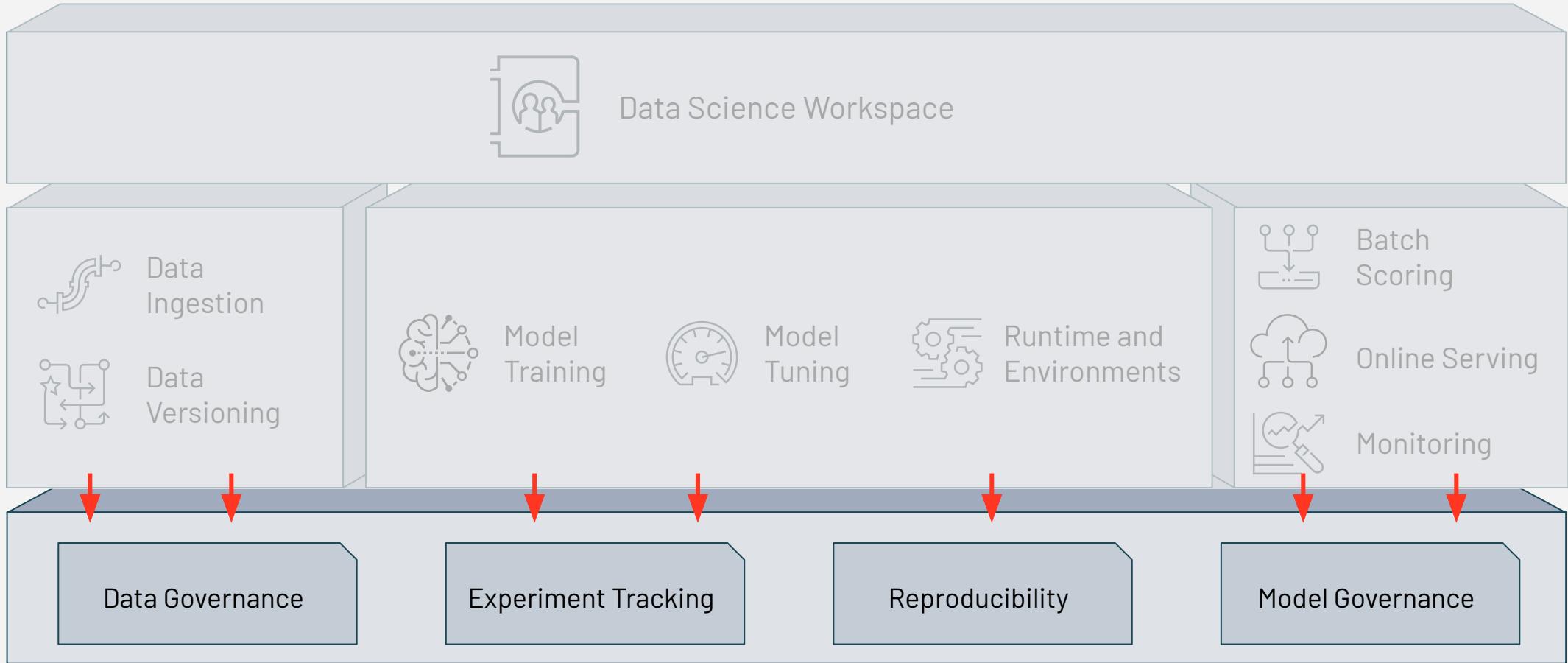


Deploying a TensorFlow  
model as a Spark UDF

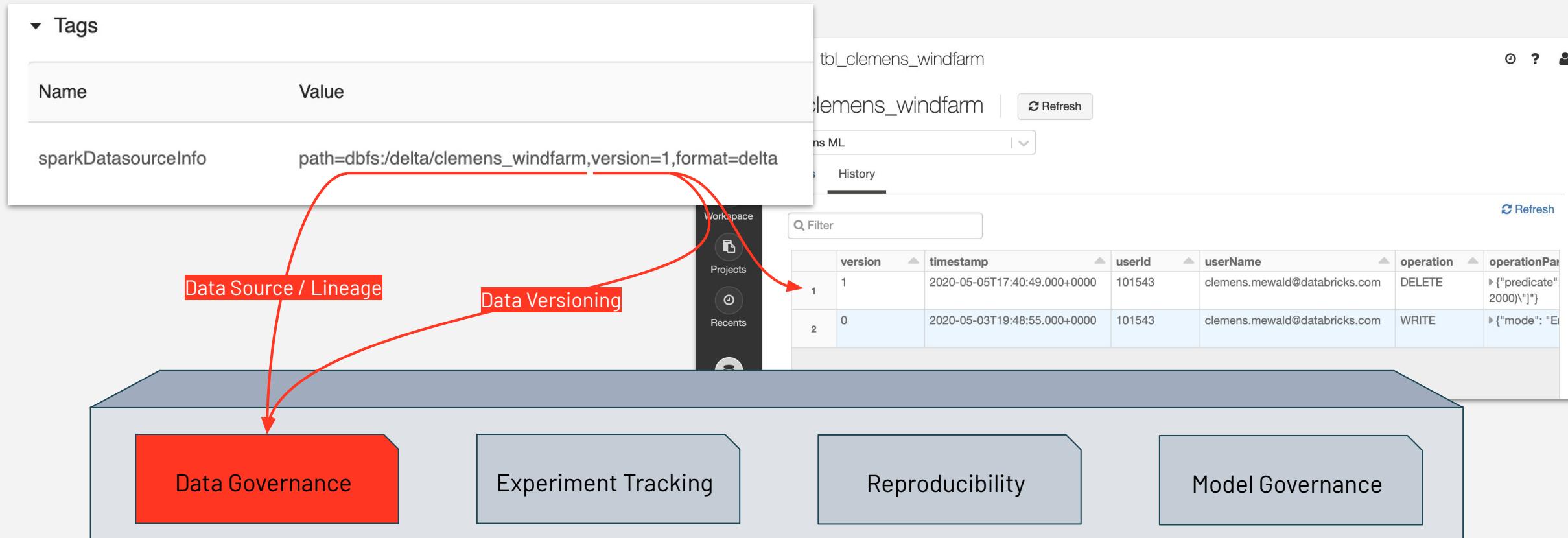
```
model_udf = mlflow.pyfunc.spark_udf(spark, model_uri='models:/clemens-windfarm-signature/production')
df.with_column('prediction', model_udf)
```

Yes, they're all the same!  
As are the commands to  
deploy these models as  
Docker containers, etc.

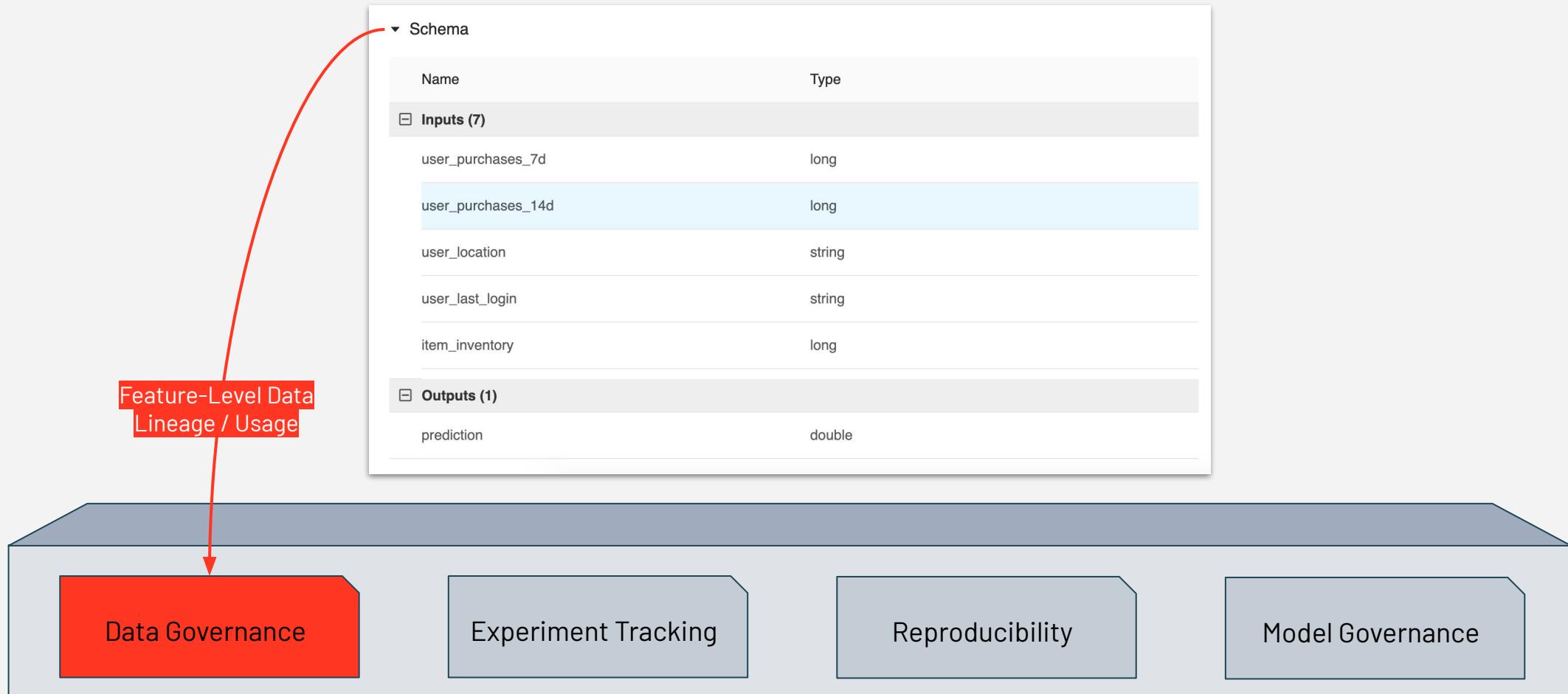
# End-to-end MLOps / Governance



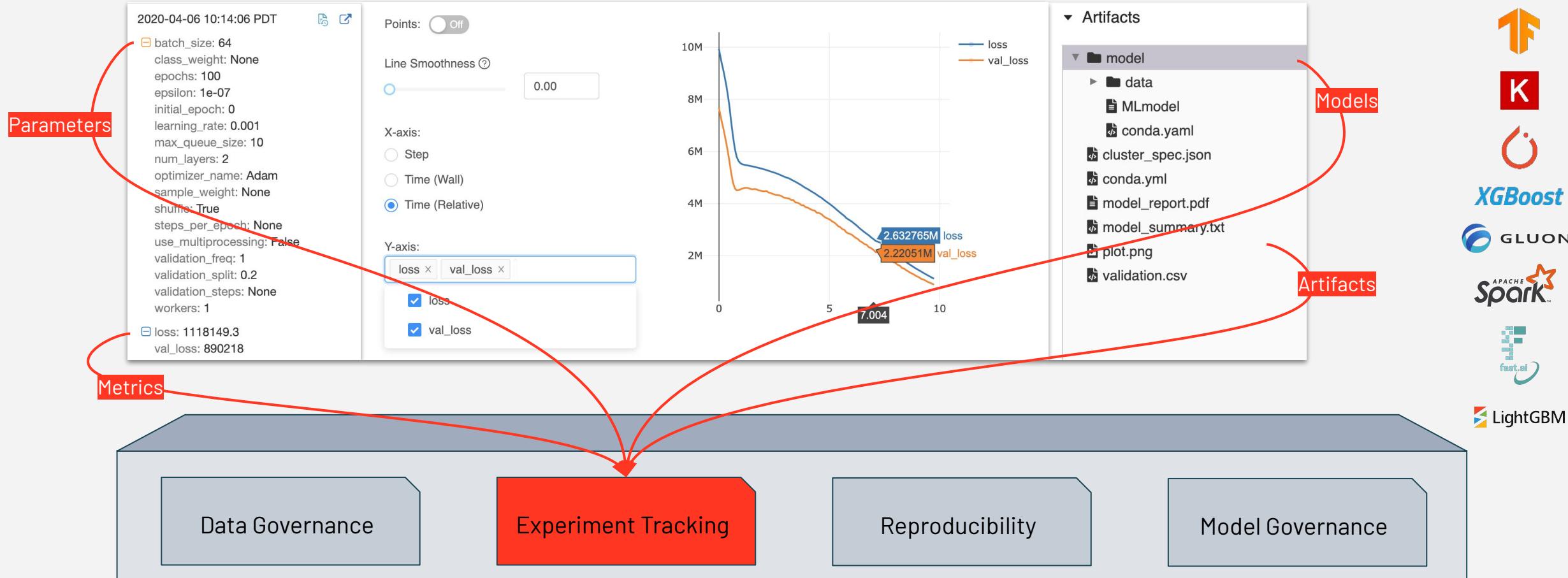
# Automated Data Source capture and Versioning



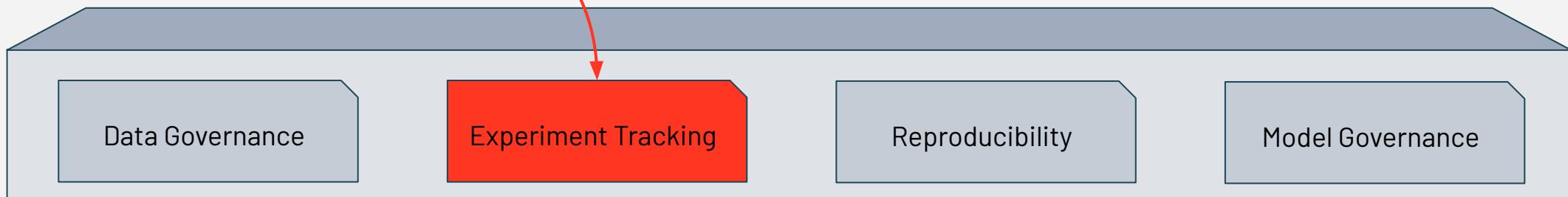
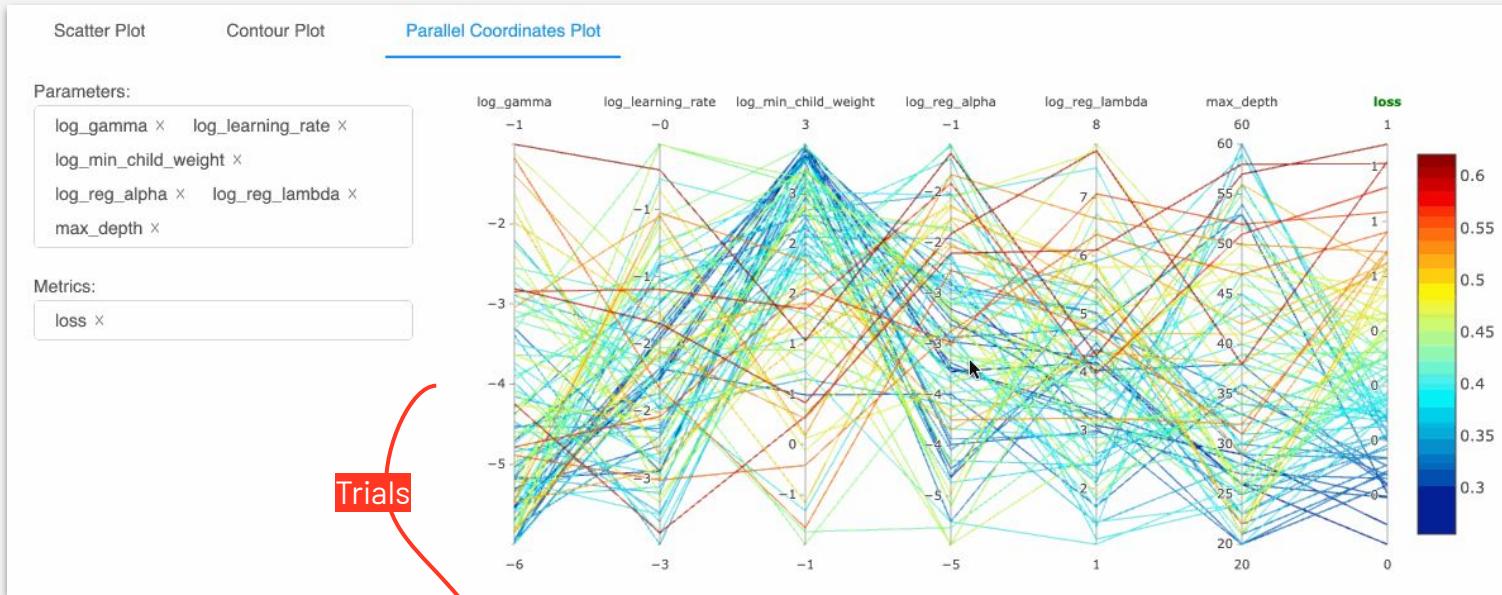
# Automated capture of Feature Usage



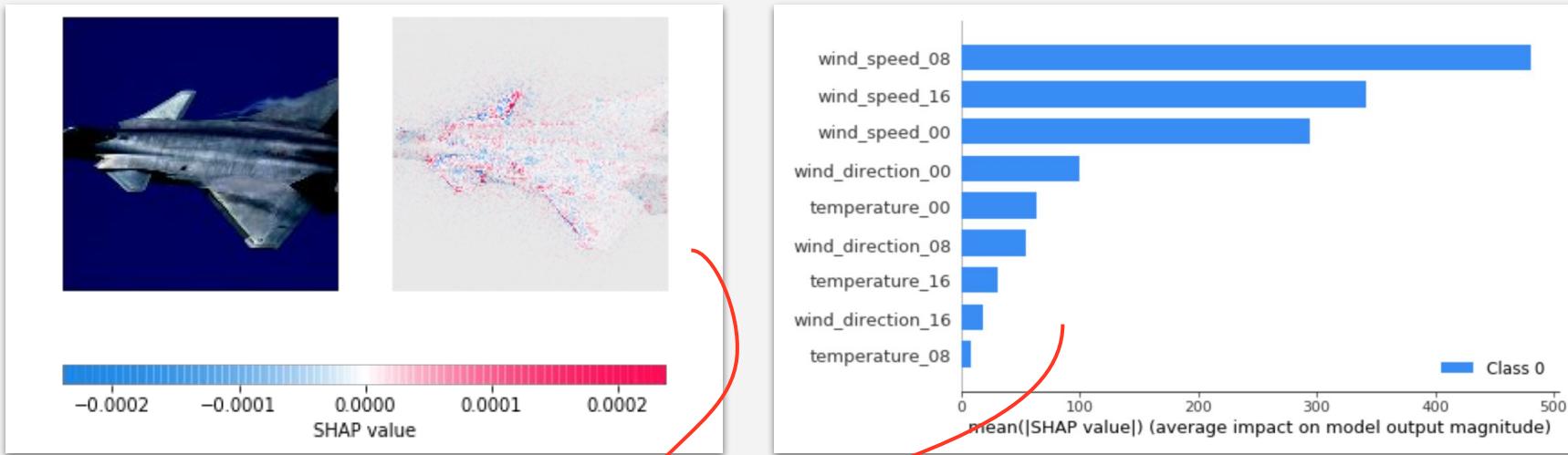
# Automated capture of ML metrics, parameters, artifacts, etc.



# Automated capture of Hyperparameter Search



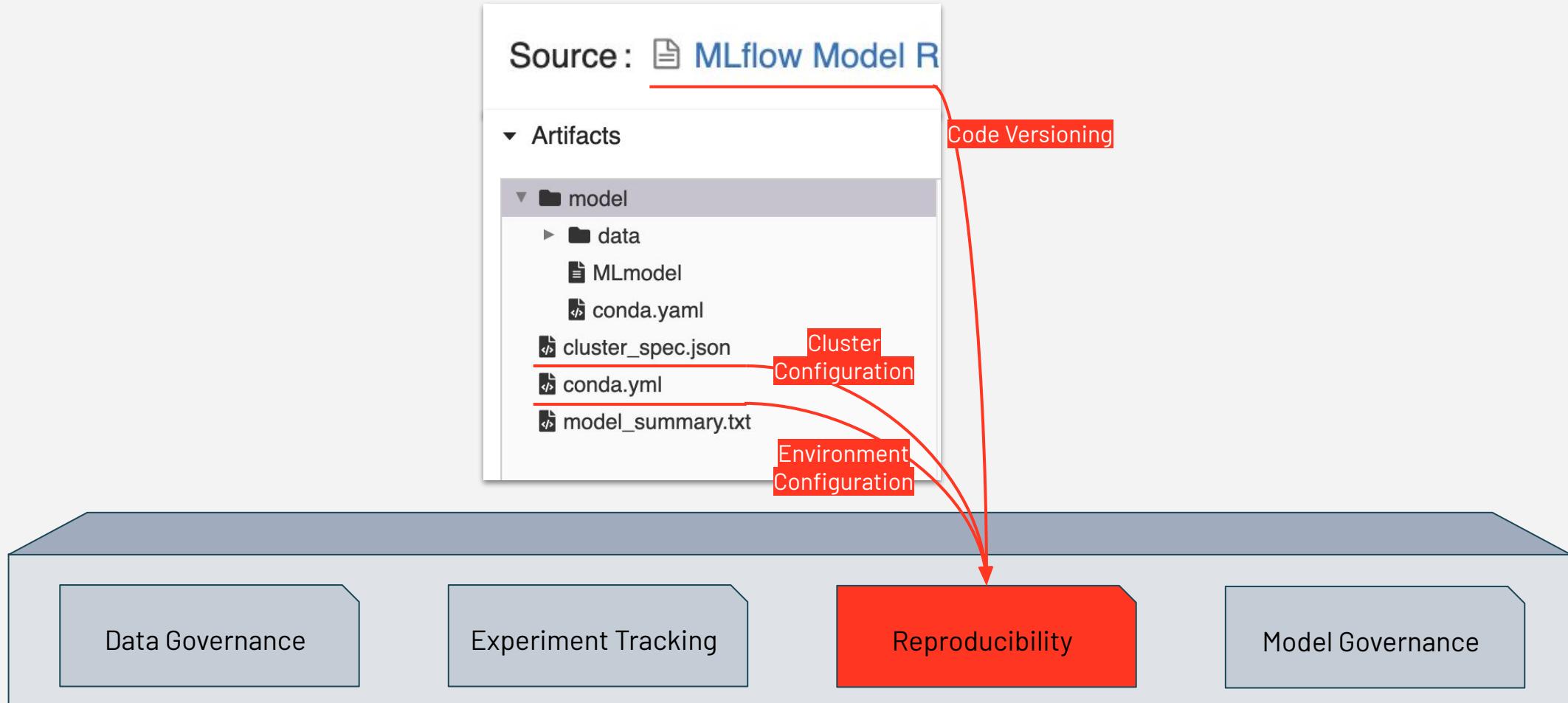
# Automated Model Interpretability



Model Interpretability



# Automated capture of Code, Environment and Cluster Specification



# Model Sharing, Reuse, and ACLs

The diagram illustrates the integration of Model Discoverability and Model Stage-Based ACLs across four key governance pillars: Data Governance, Experiment Tracking, Reproducibility, and Model Governance.

**Model Discoverability** (represented by a red box) is shown as a search bar in the Registered Models interface, allowing users to find specific models like "clemens-model2".

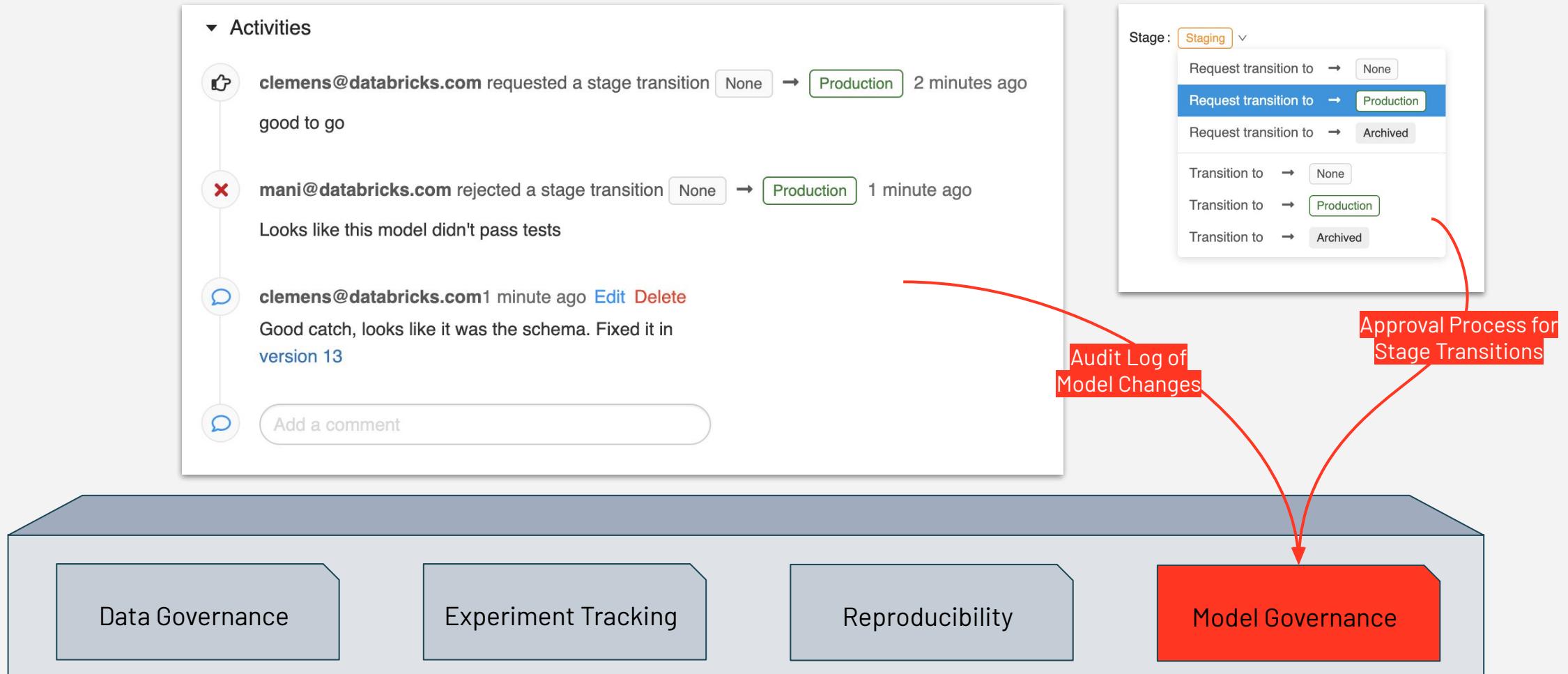
**Model Stage-Based ACLs** (represented by a red box) is shown in the Permission Settings dialog, where "clemens.mewald@databricks.com" is granted "Can Manage" permissions for Staging Versions. A red arrow points from this dialog to the Model Governance pillar.

Name	Permission
admins	Can Manage inherited
clemens.mewald@databricks.com	Can Manage
Select User or Group...	Can Read
Select User or Group...	Can Edit
Select User or Group...	Can Manage Staging Versions
Select User or Group...	Can Manage Production Versions
Select User or Group...	Can Manage

**Data Governance**, **Experiment Tracking**, and **Reproducibility** are represented by light blue boxes and are grouped under the Model Discoverability pillar.

**Model Governance** is represented by a red box and is grouped under the Model Stage-Based ACLs pillar.

# Automated Model Lineage and Governance



# Turnkey Model Serving

The screenshot shows the Databricks Registered Models interface for the model 'clemens-windfarm-signature'. The 'Serving' tab is selected. The status is 'Ready - Stop'. The cluster used is 'mlflow-model-clemens-windfarm-signature'. The 'Model Versions' section lists five versions:

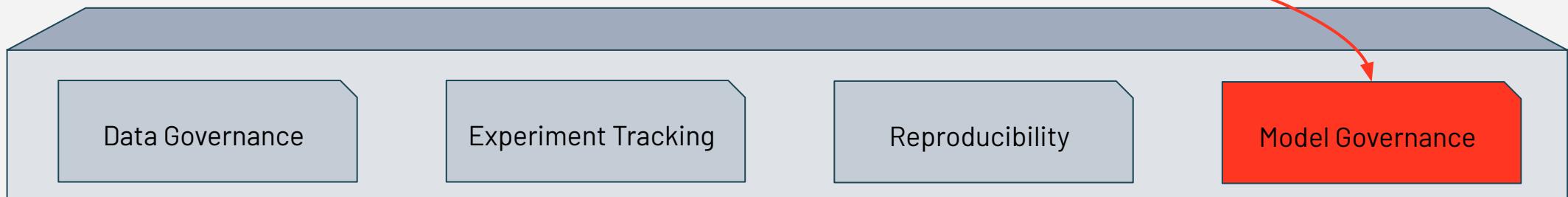
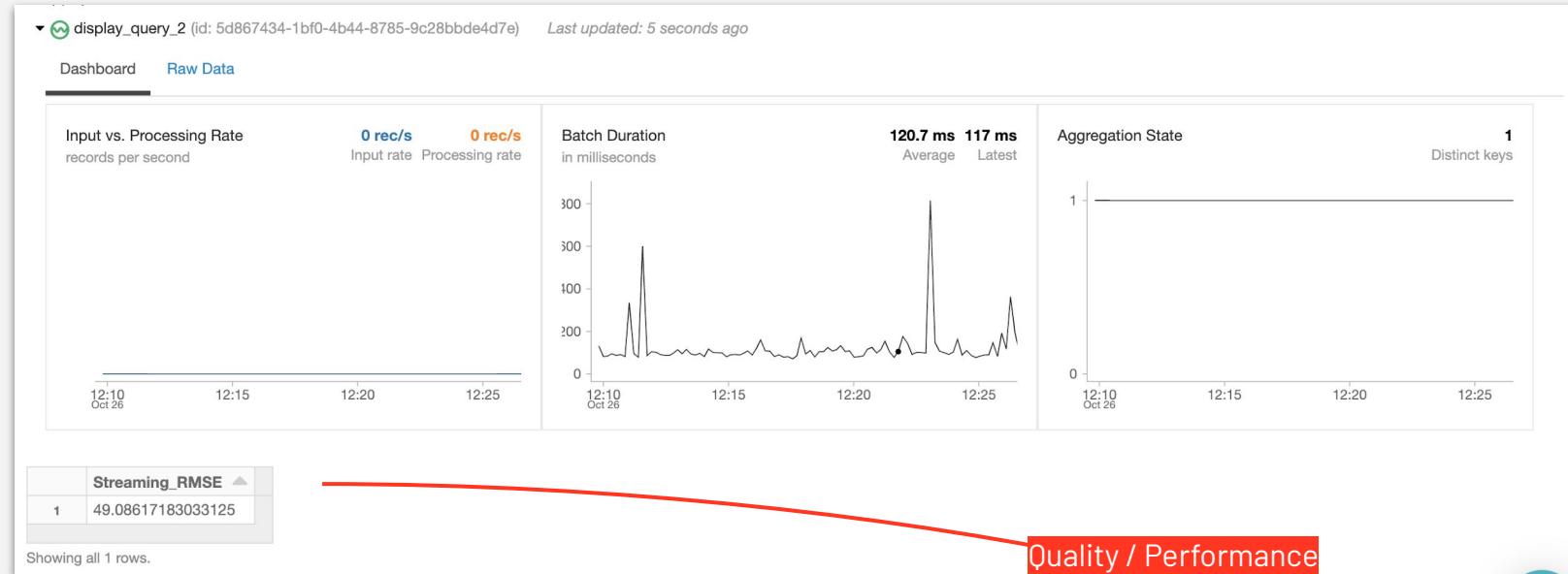
Version	Status
Version 1	Ready
Version 2	Staging
Version 3	Ready
Version 4	Ready
Version 5	Ready

Model URLs are provided for each version. The 'Call the model' section shows a request and response. A red box highlights the integration between Turnkey Serving and Model Versions/Stages.

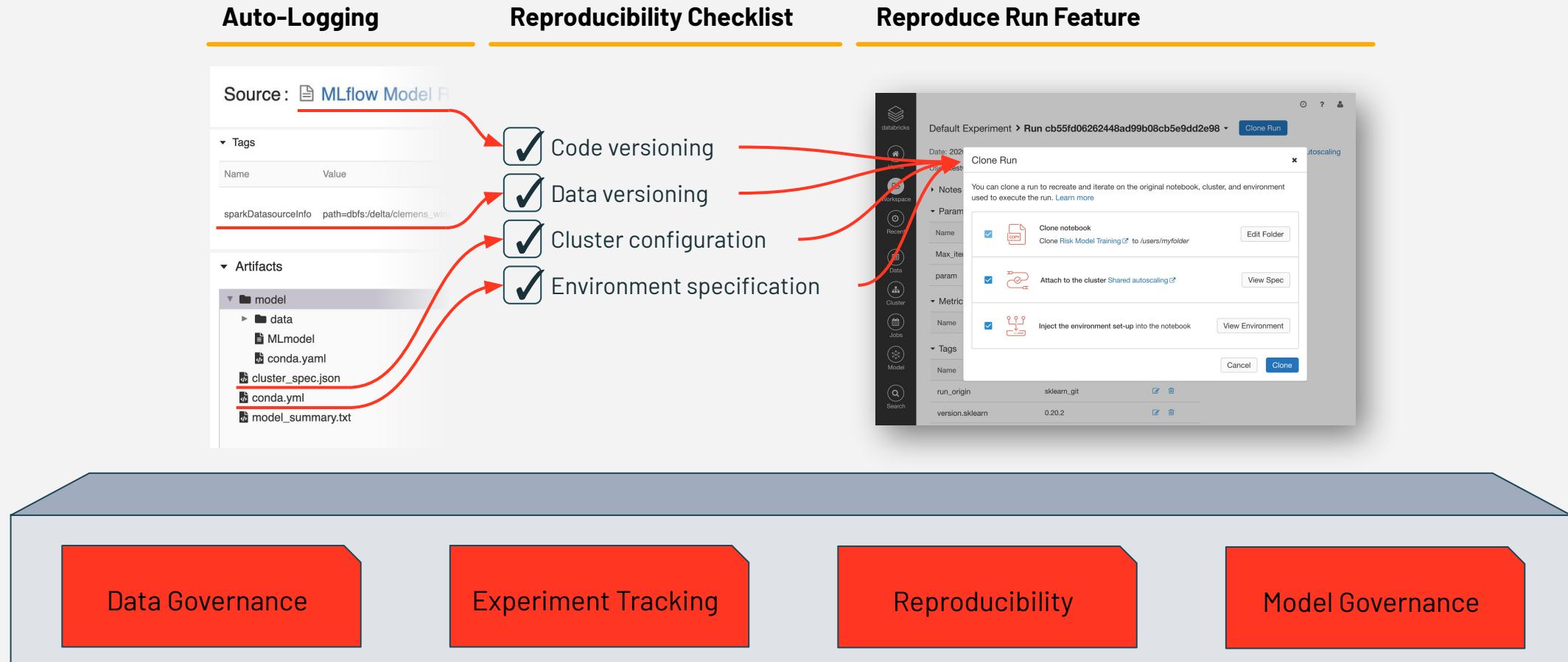
Turnkey Serving integrated with Model Versions and Stages

Data Governance      Experiment Tracking      Reproducibility      Model Governance

# Model Quality monitoring



# The Result: Full End-to-End Governance and Reproducibility



# The Databricks ML Platform

