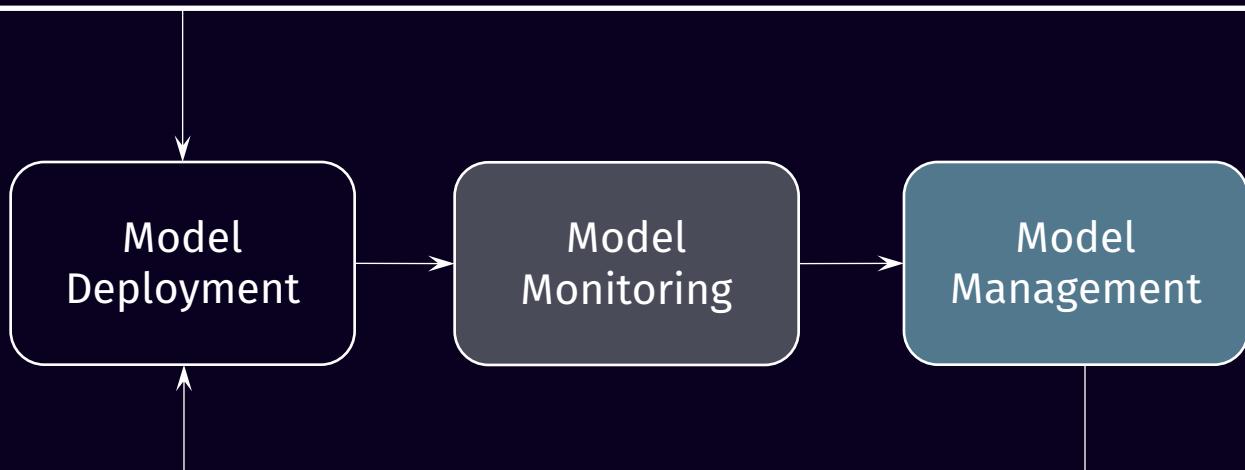
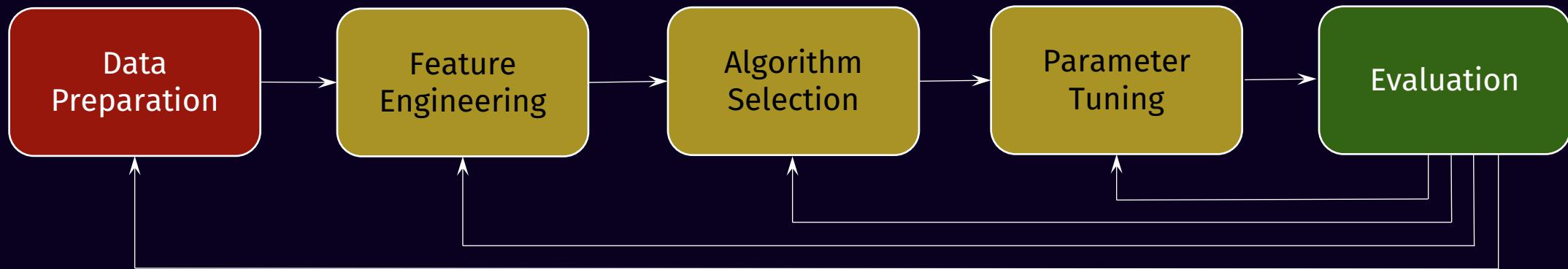


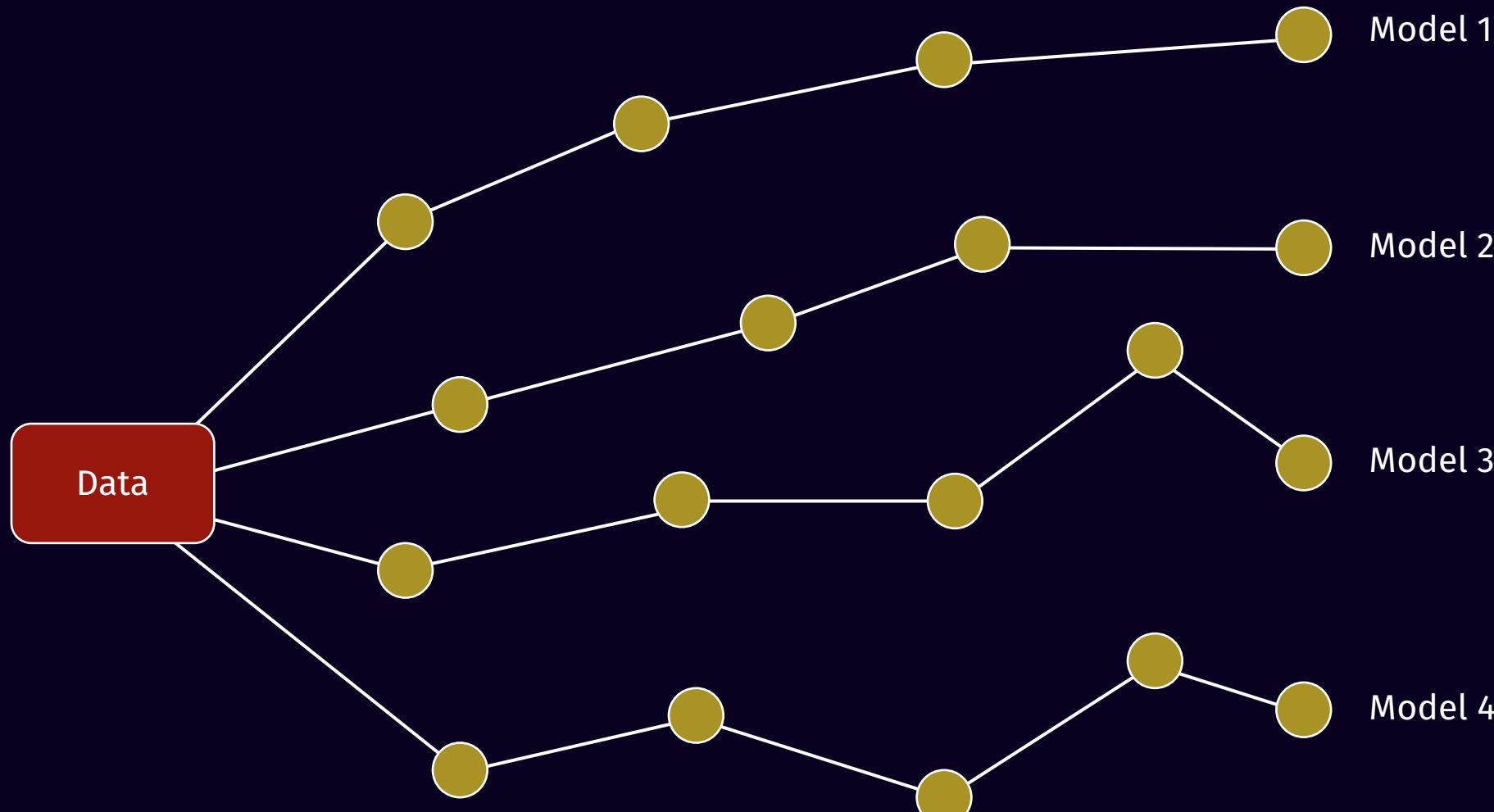


Towards Automated Tracking of Machine Learning Experiments

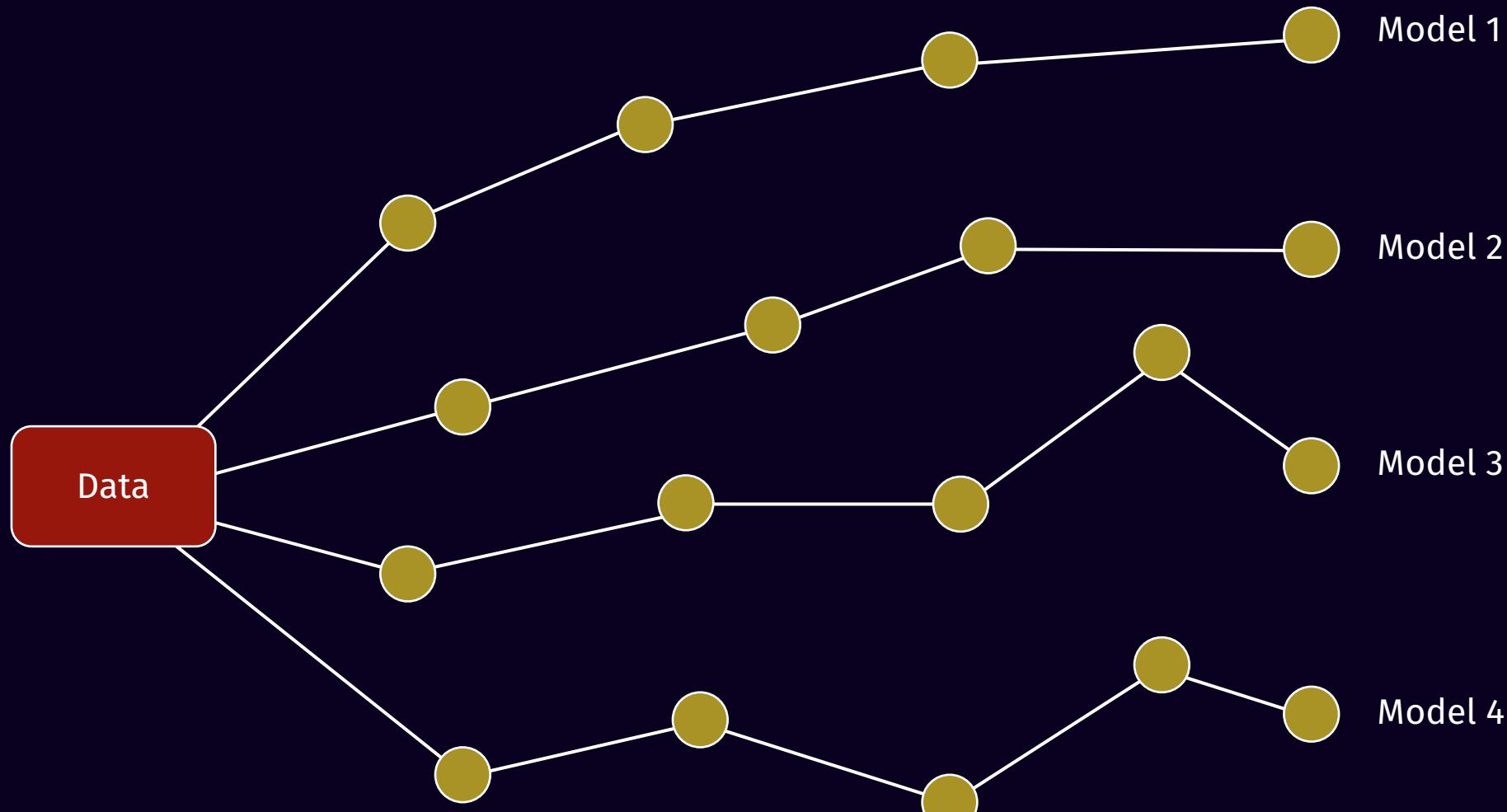
Joos-Hendrik Boese | Johannes Kirschnick | Thoralf Klein | **Stephan Seufert**
Ryck ExperimentTracker | Core AI

Model Selection

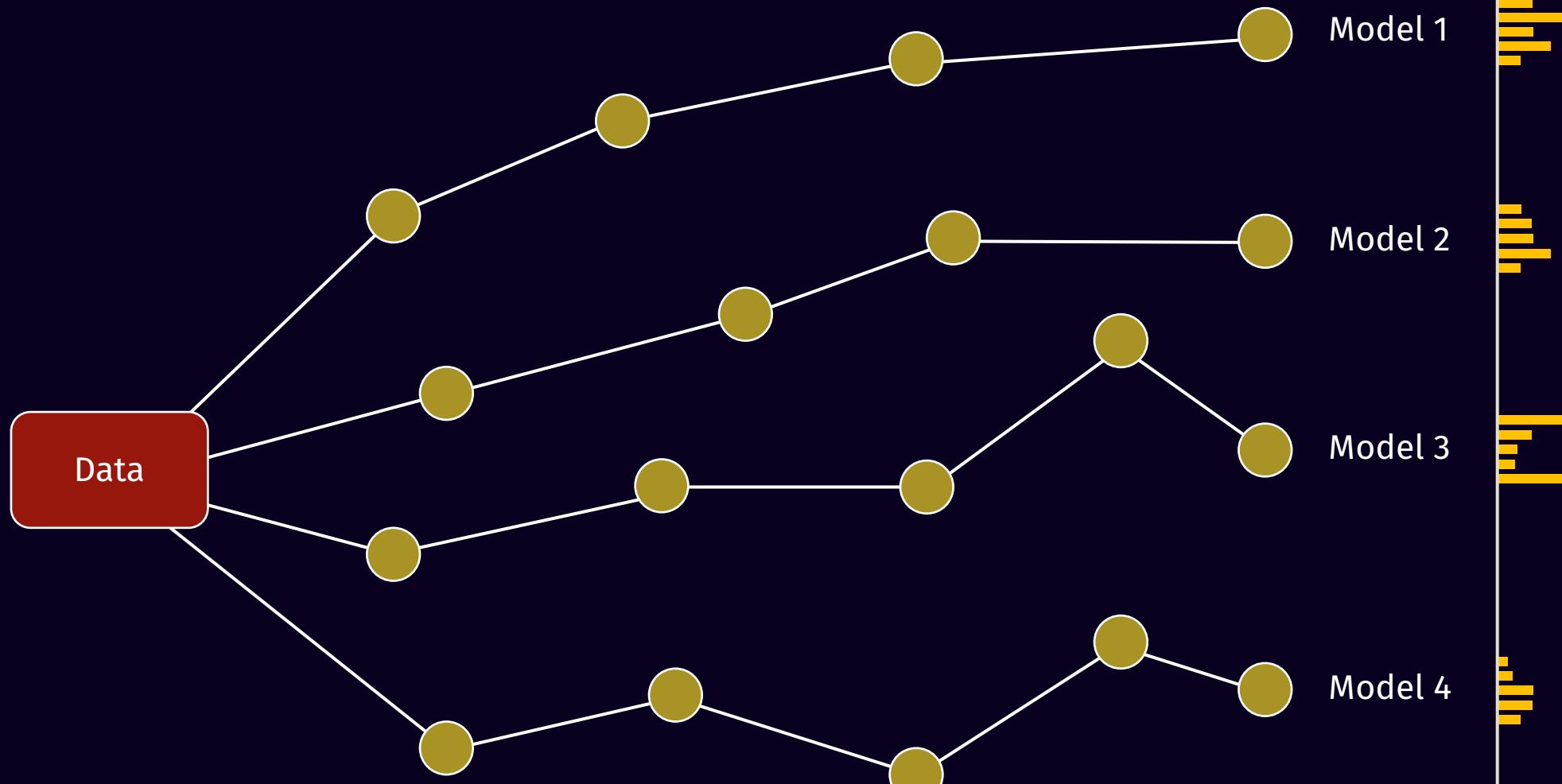
Experimentation



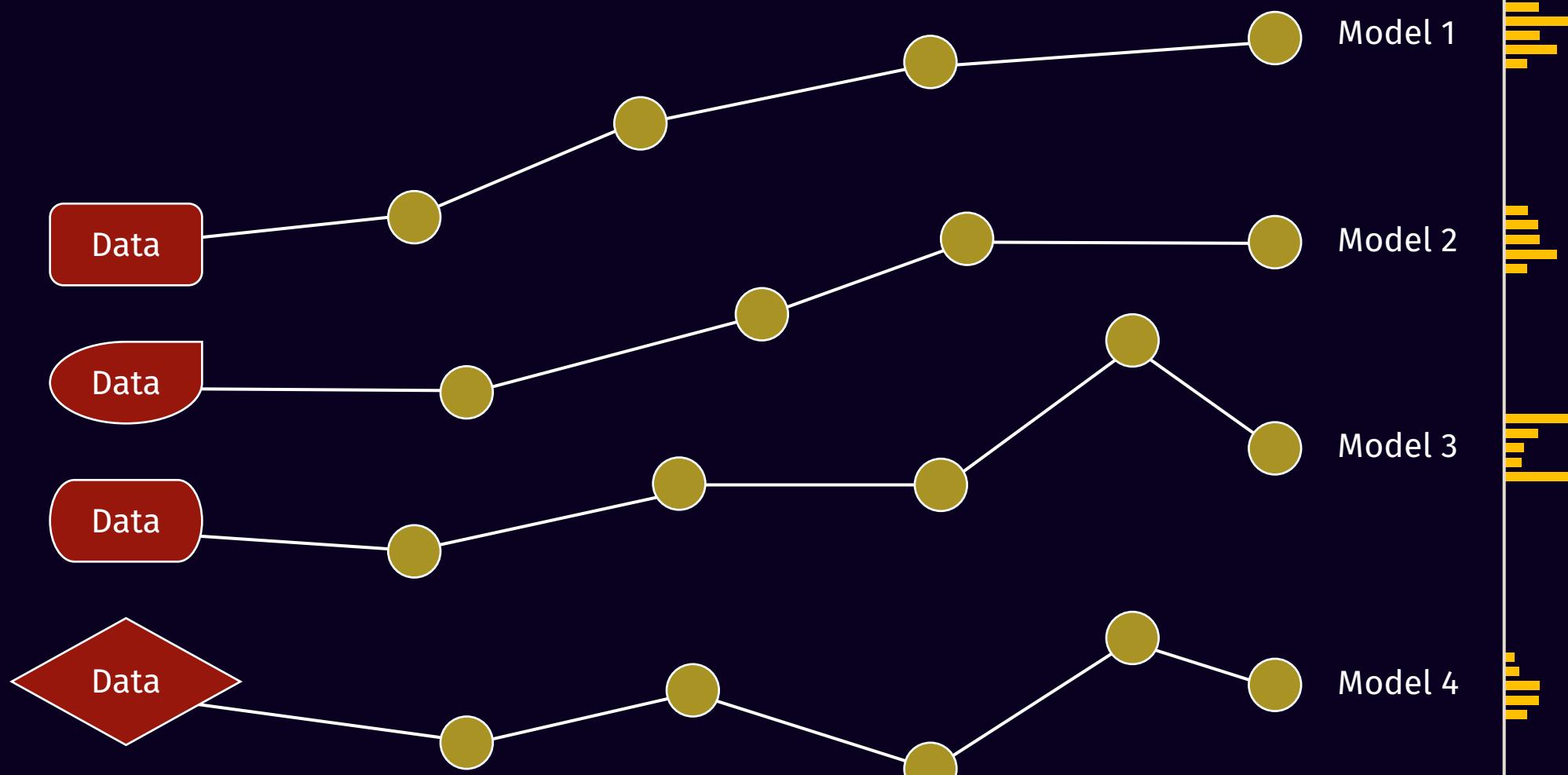
Experimentation



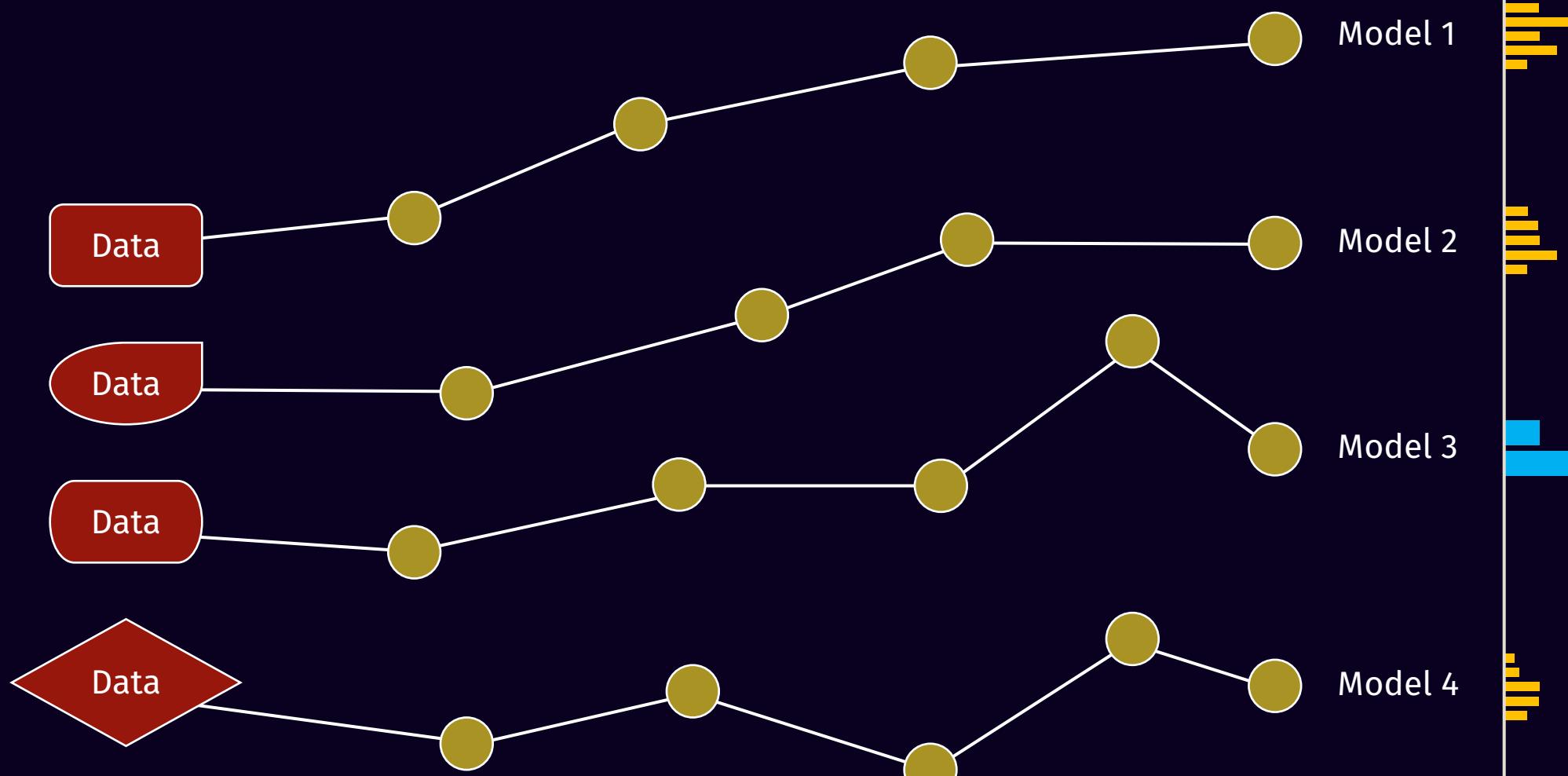
Experimentation



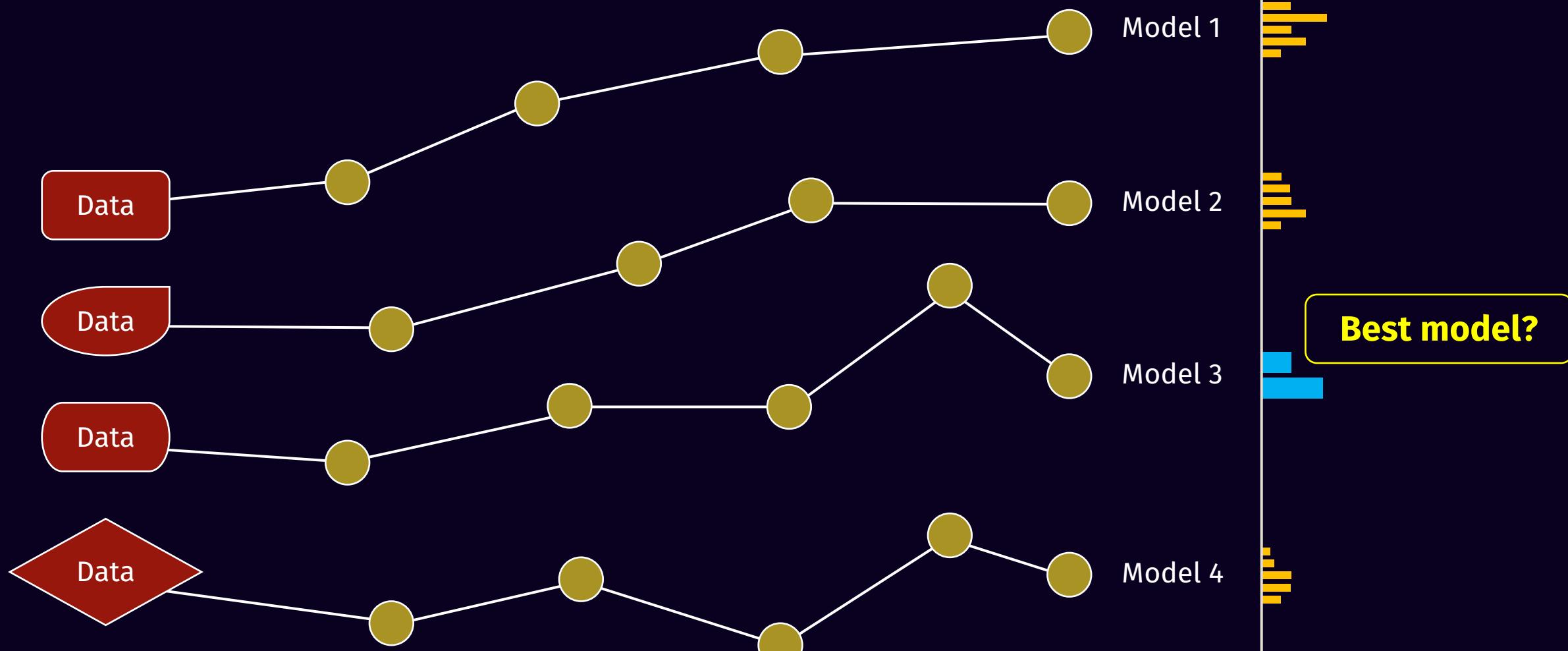
Experimentation



Experimentation

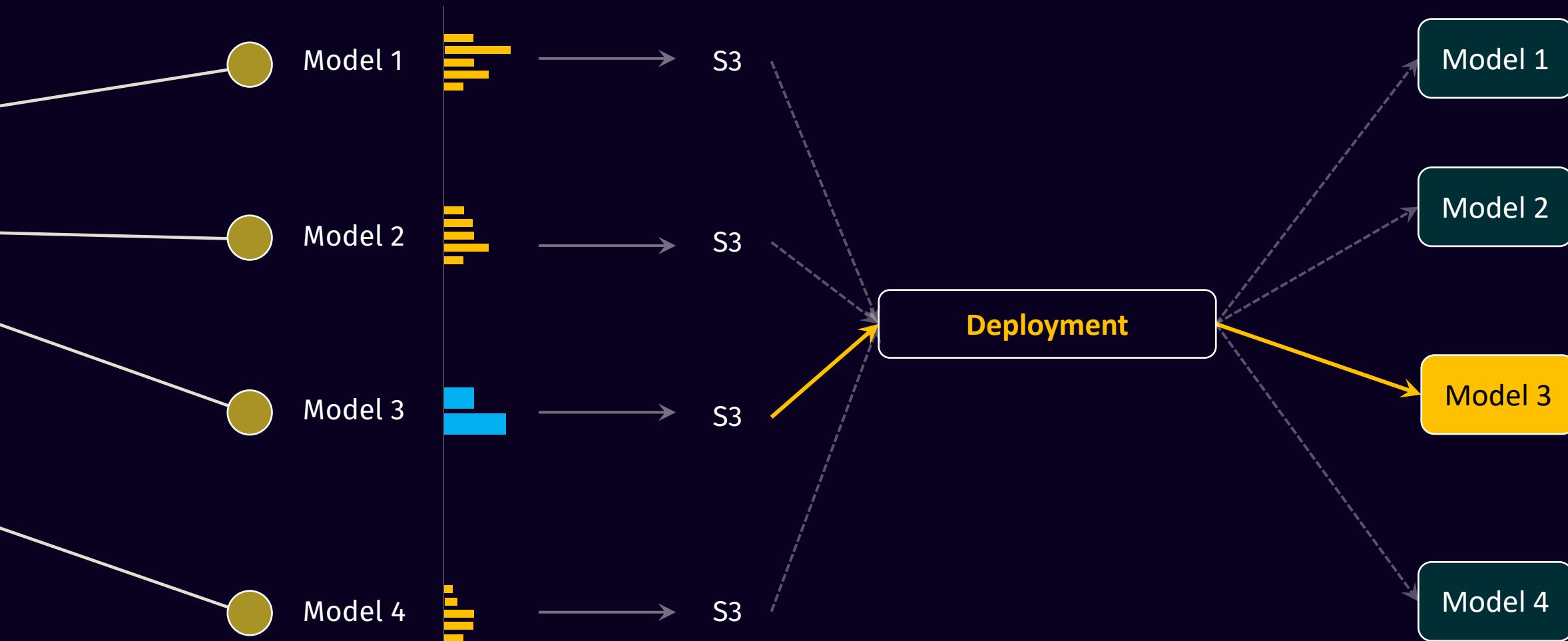


Experimentation



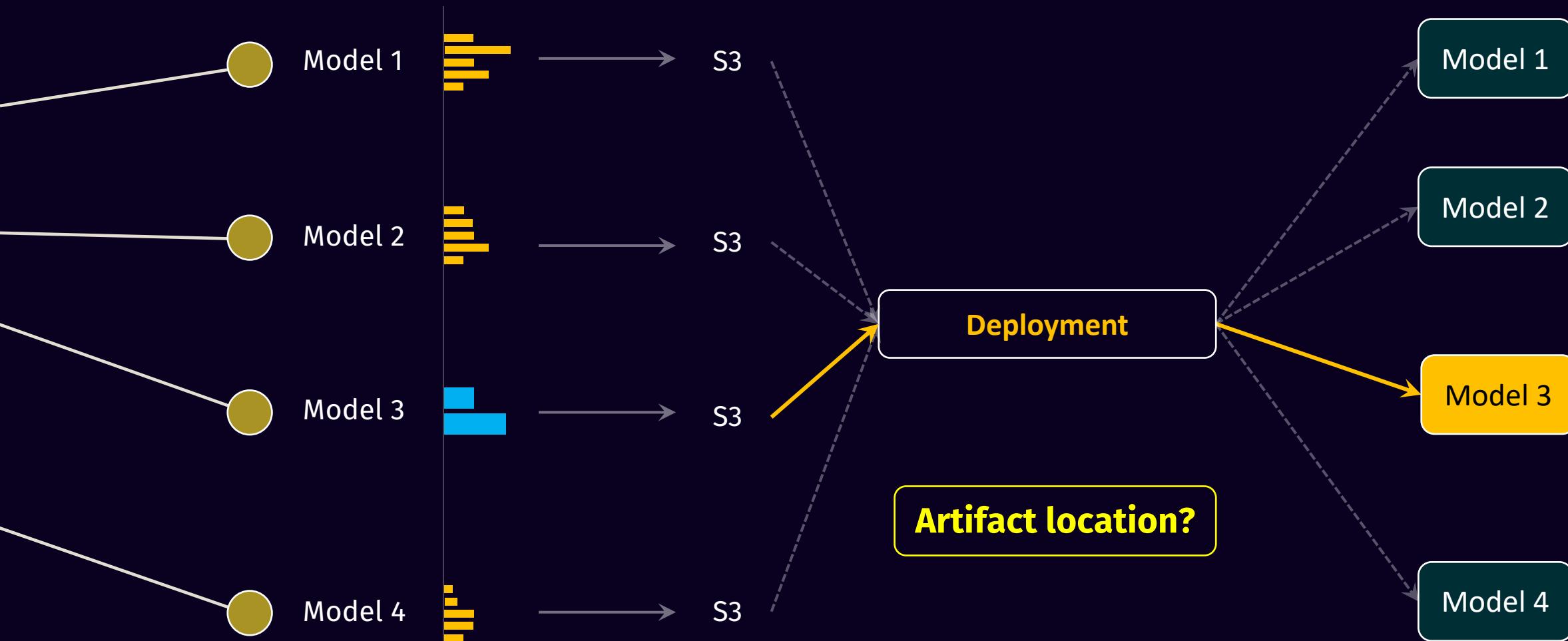
Experimentation

Production

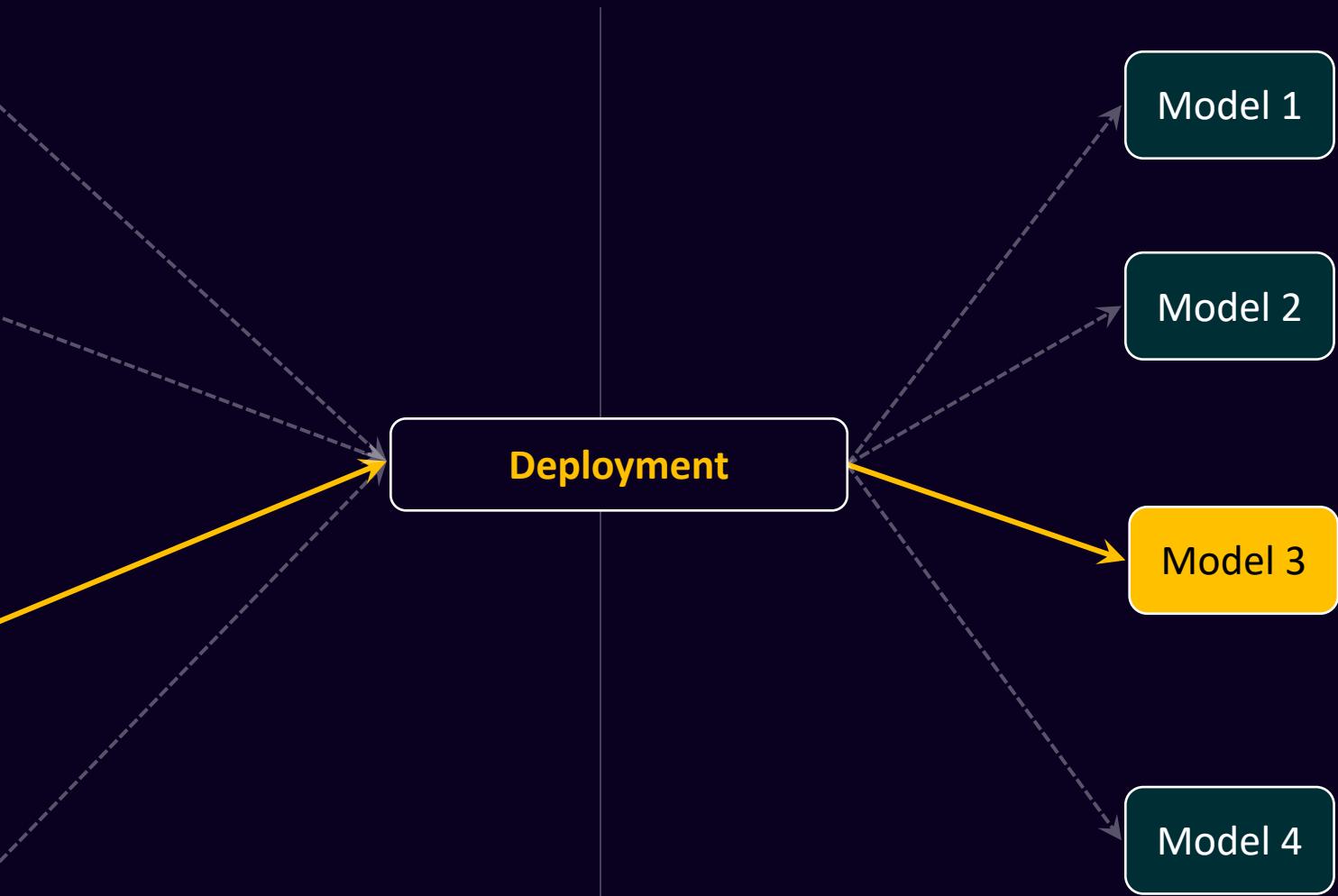


Experimentation

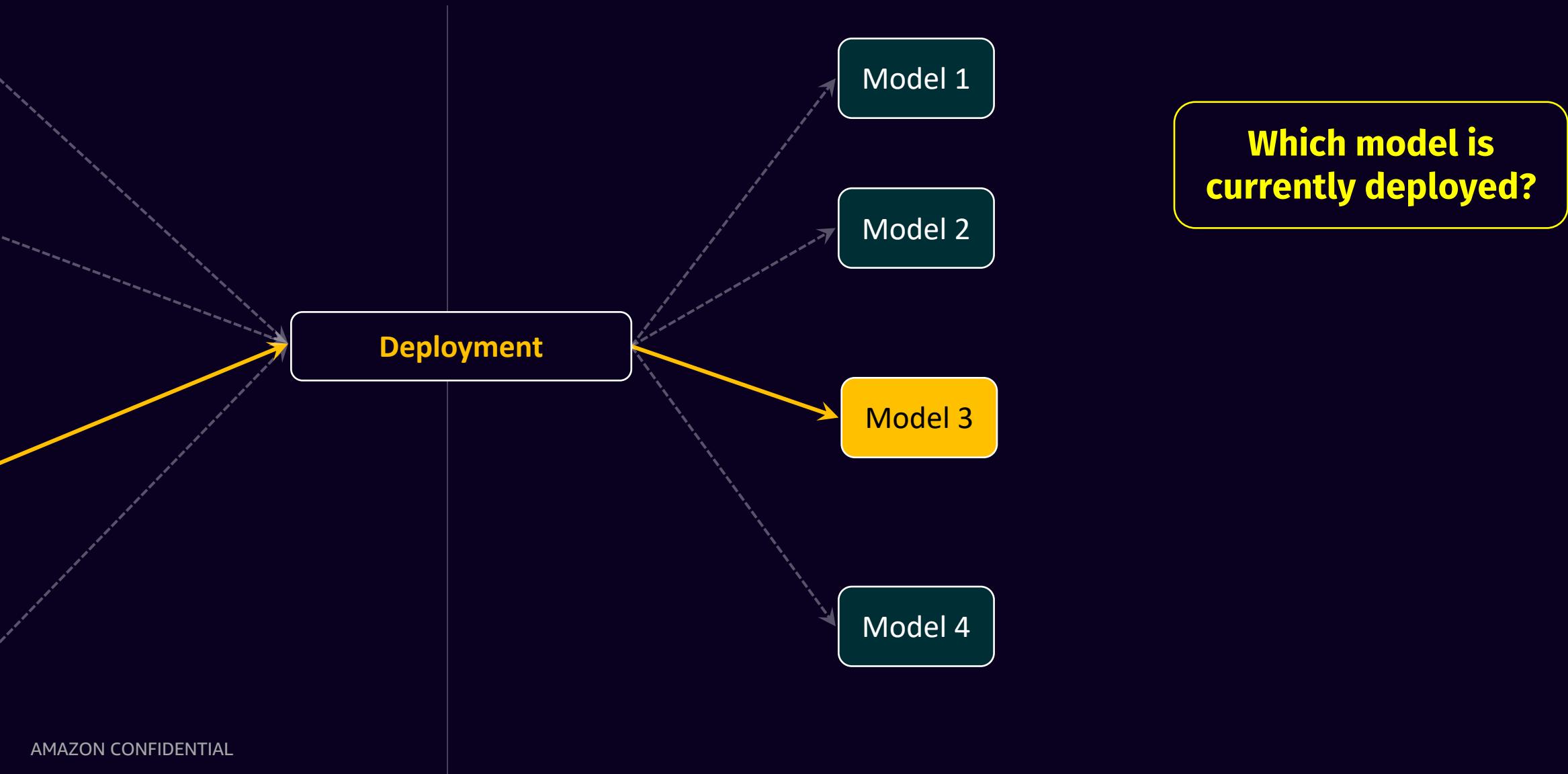
Production



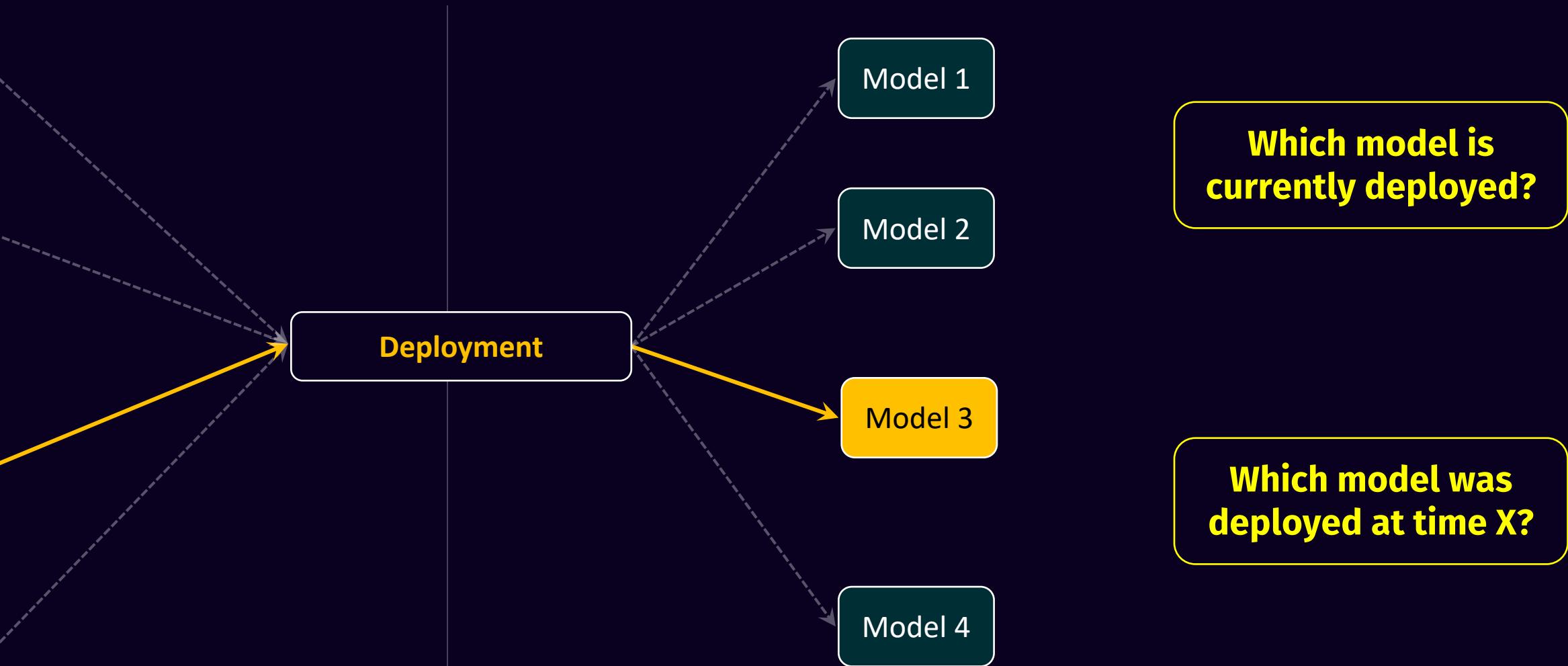
Production



Production

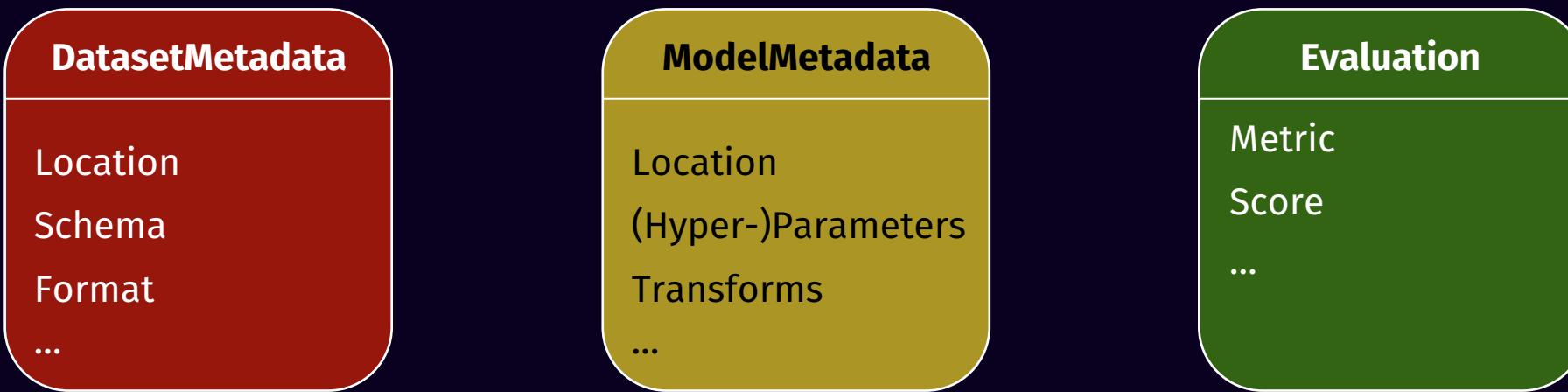


Production

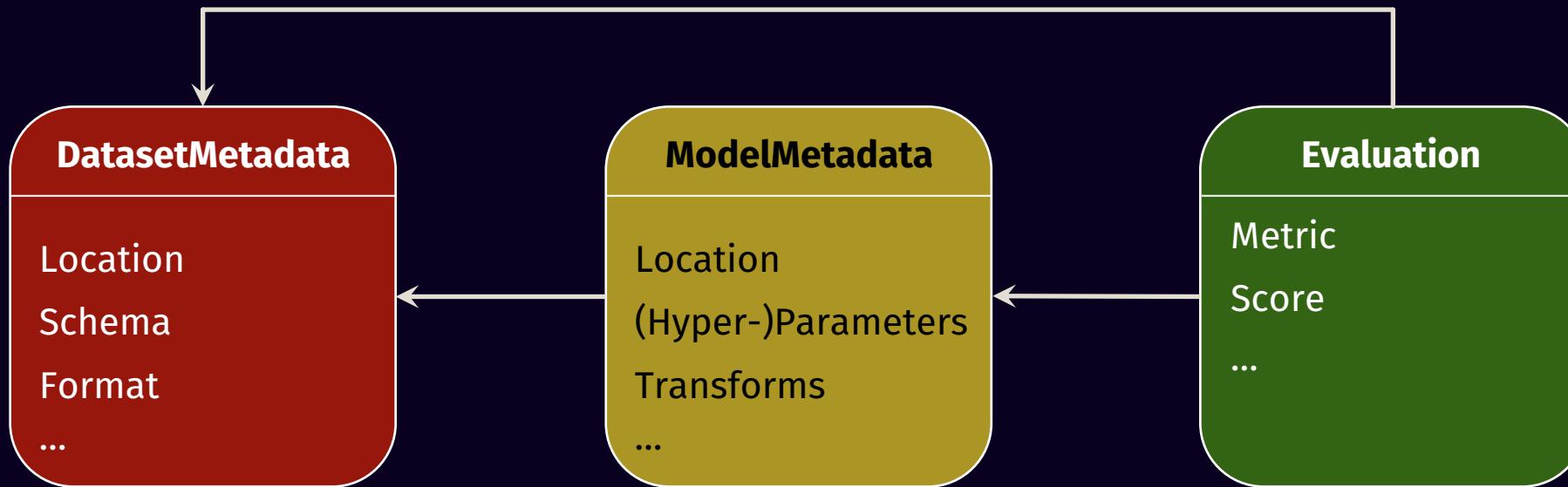


ExperimentTracker

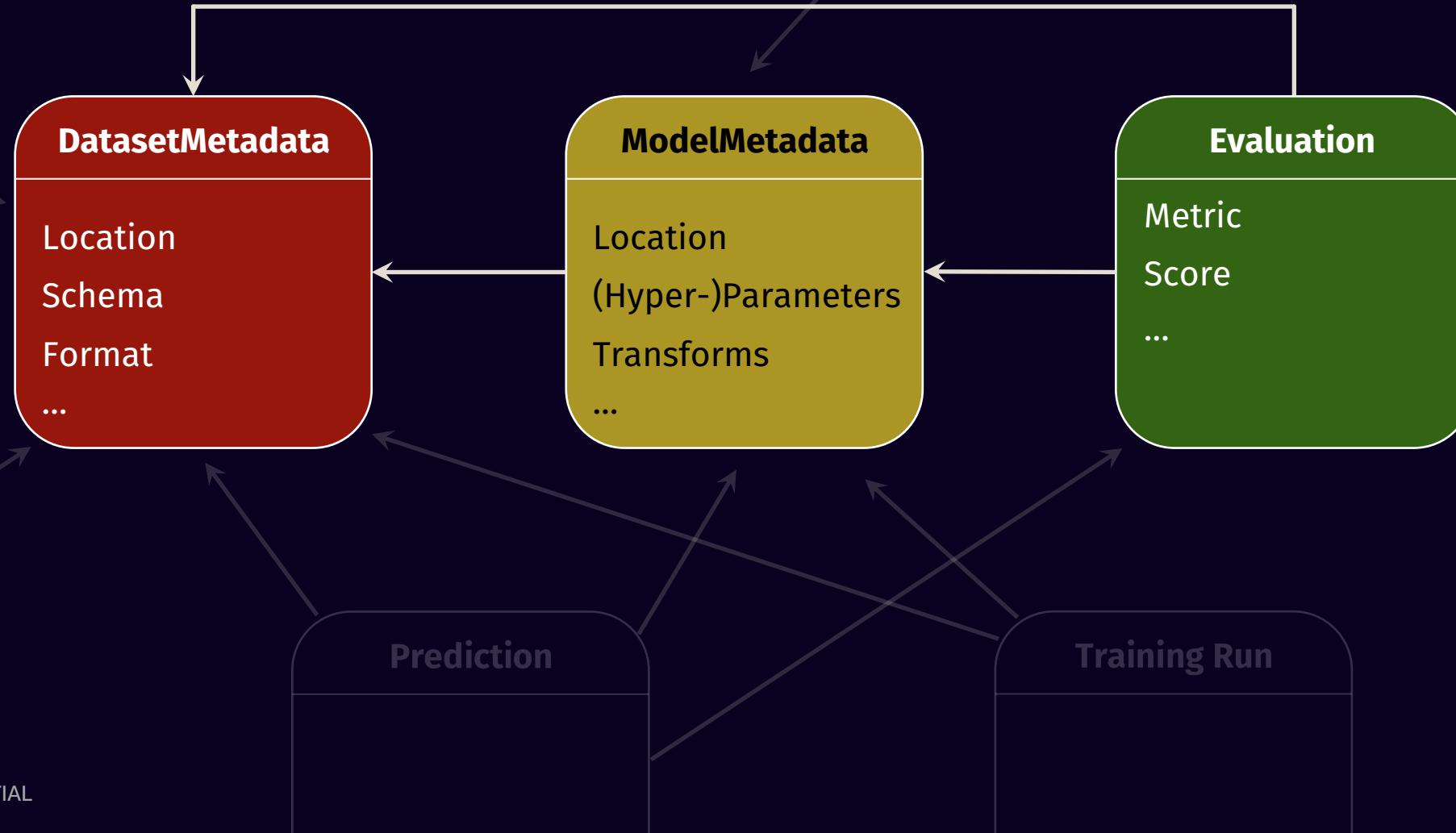
Metadata Schema (excerpt)



Metadata Schema (excerpt)

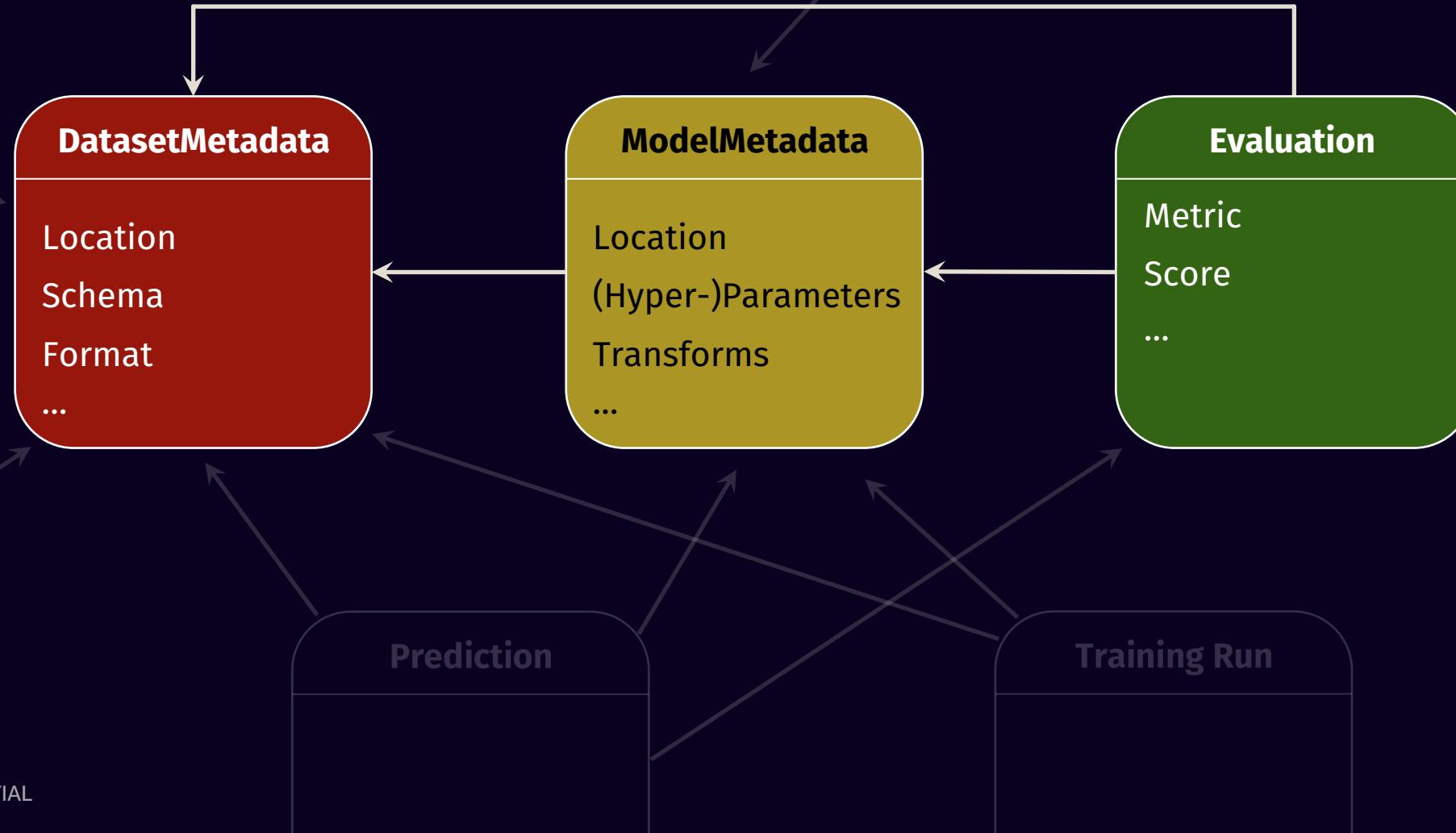


Metadata Schema (excerpt)



Metadata Schema (excerpt)

<https://github.com/awslabs/ml-experiments-schema>

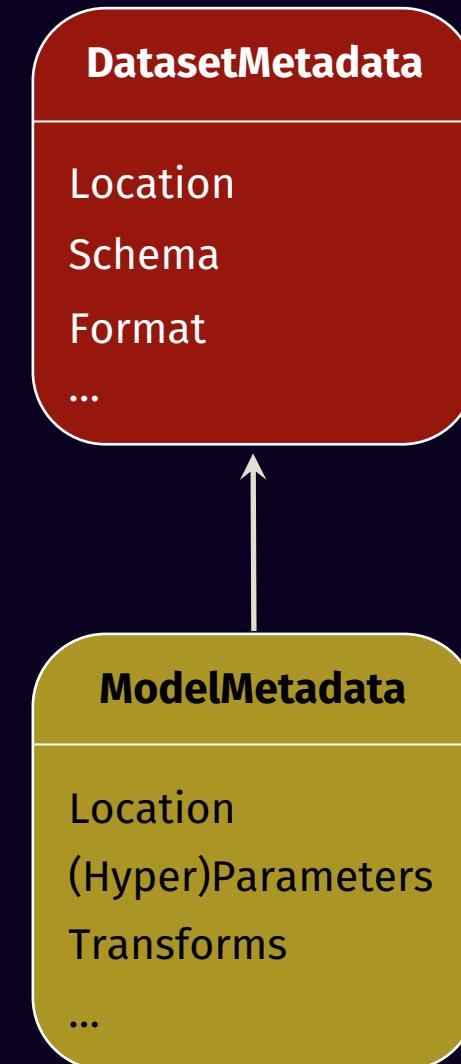


Automated Metadata Extraction

```
val df: DataFrame = ...  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)
```

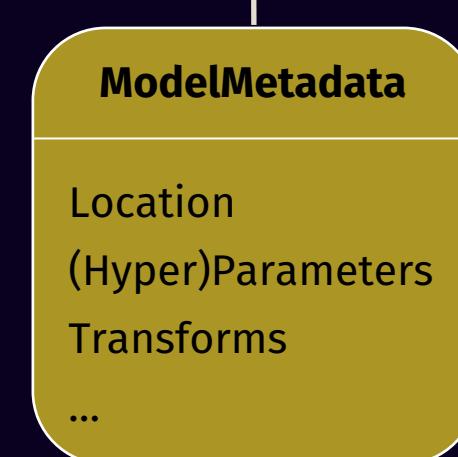
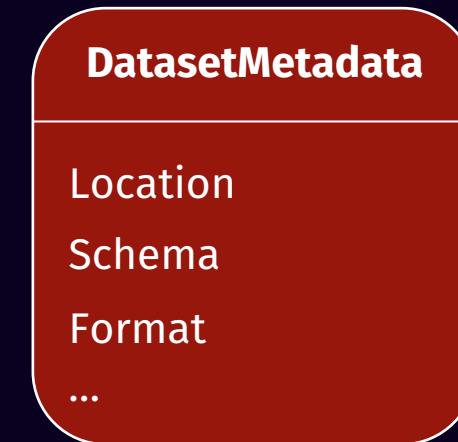
Automated Metadata Extraction

```
val df: DataFrame = ...  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)
```



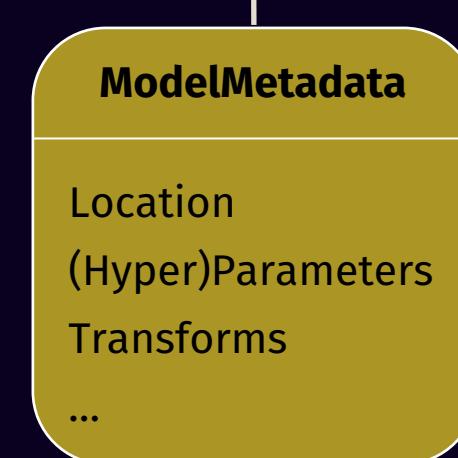
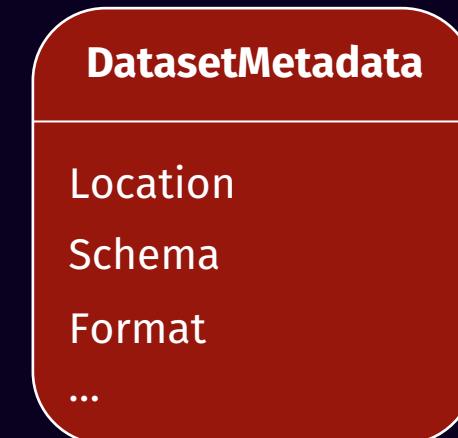
Automated Metadata Extraction

```
val df: DataFrame = ...  
tracker.trackDataset(df)  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)  
tracker.trackModel(model)
```

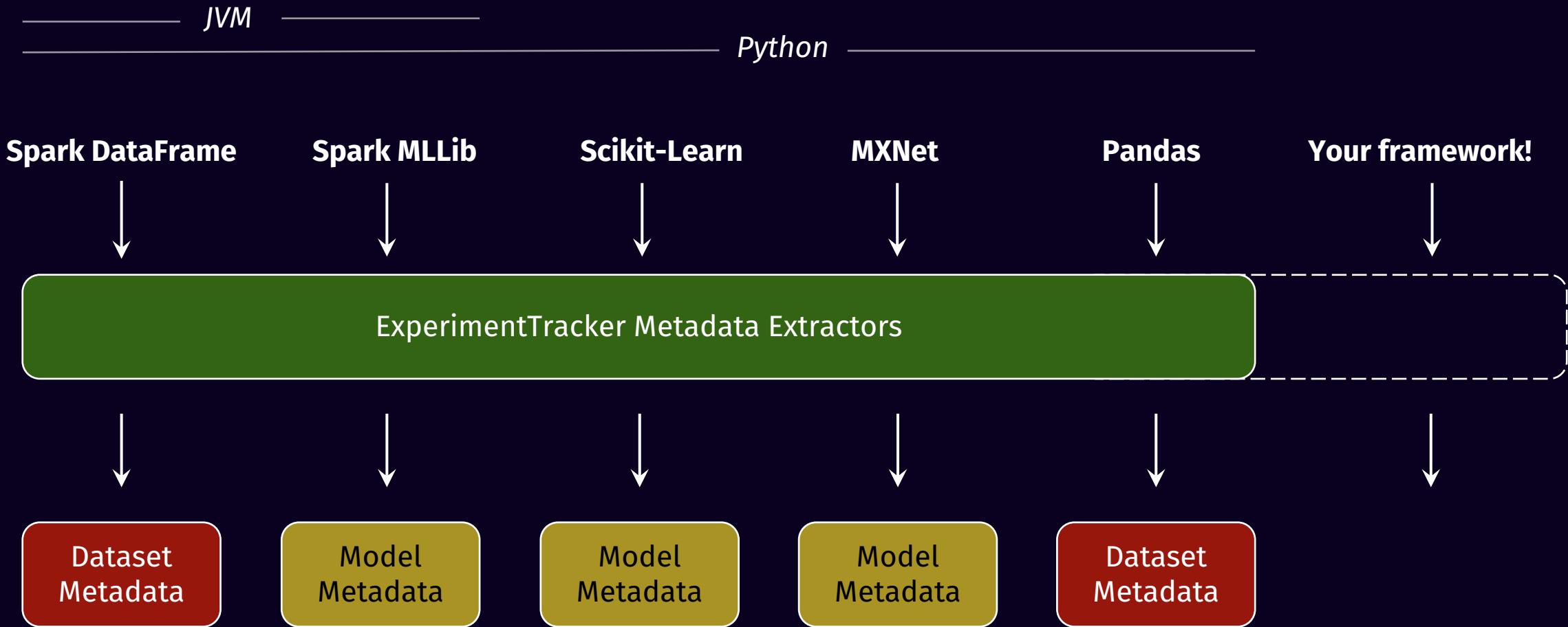


Automated Metadata Extraction

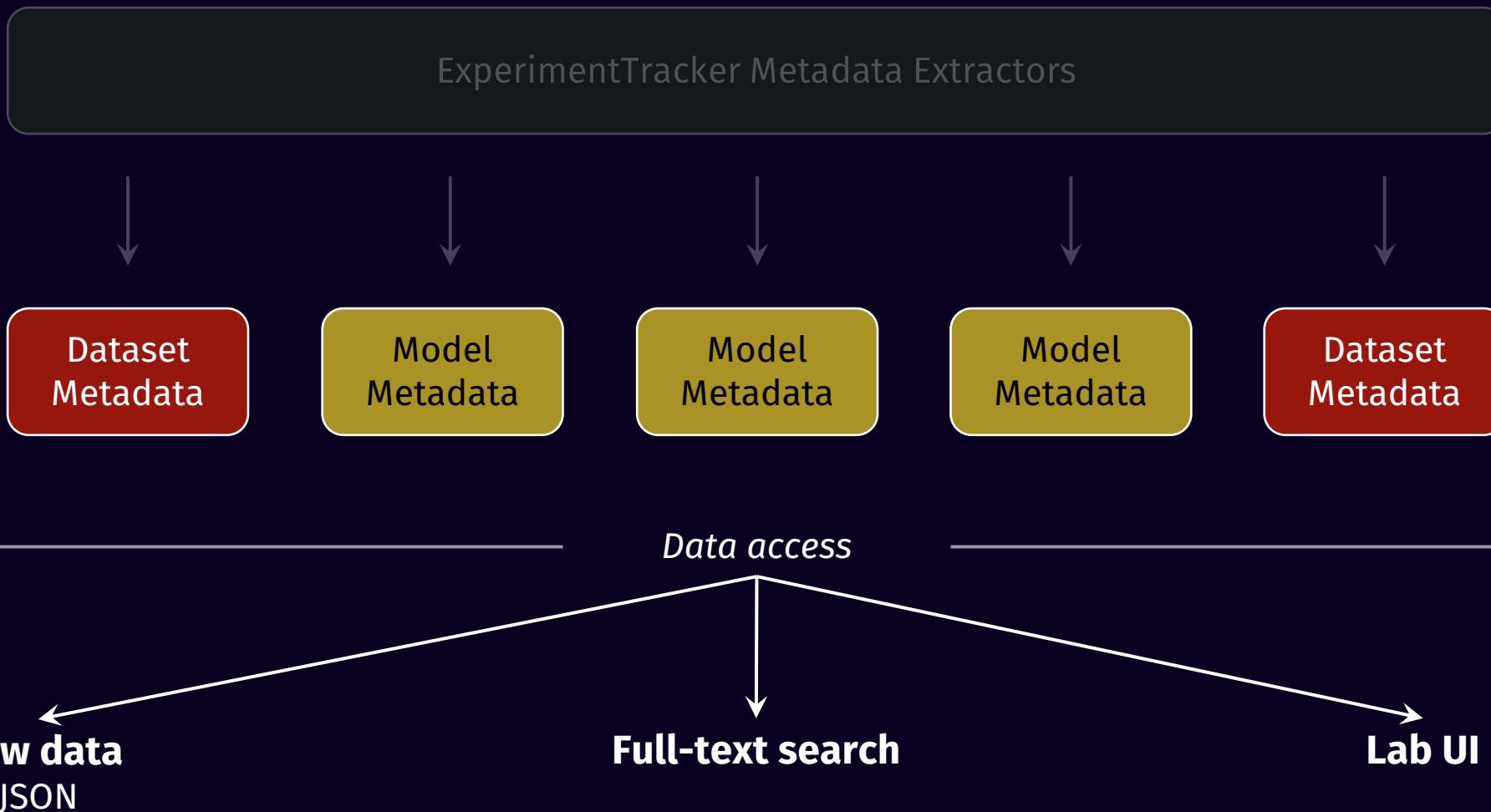
```
val df: DataFrame = ...  
tracker.trackDataset(df)  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)  
tracker.trackModel(model)
```



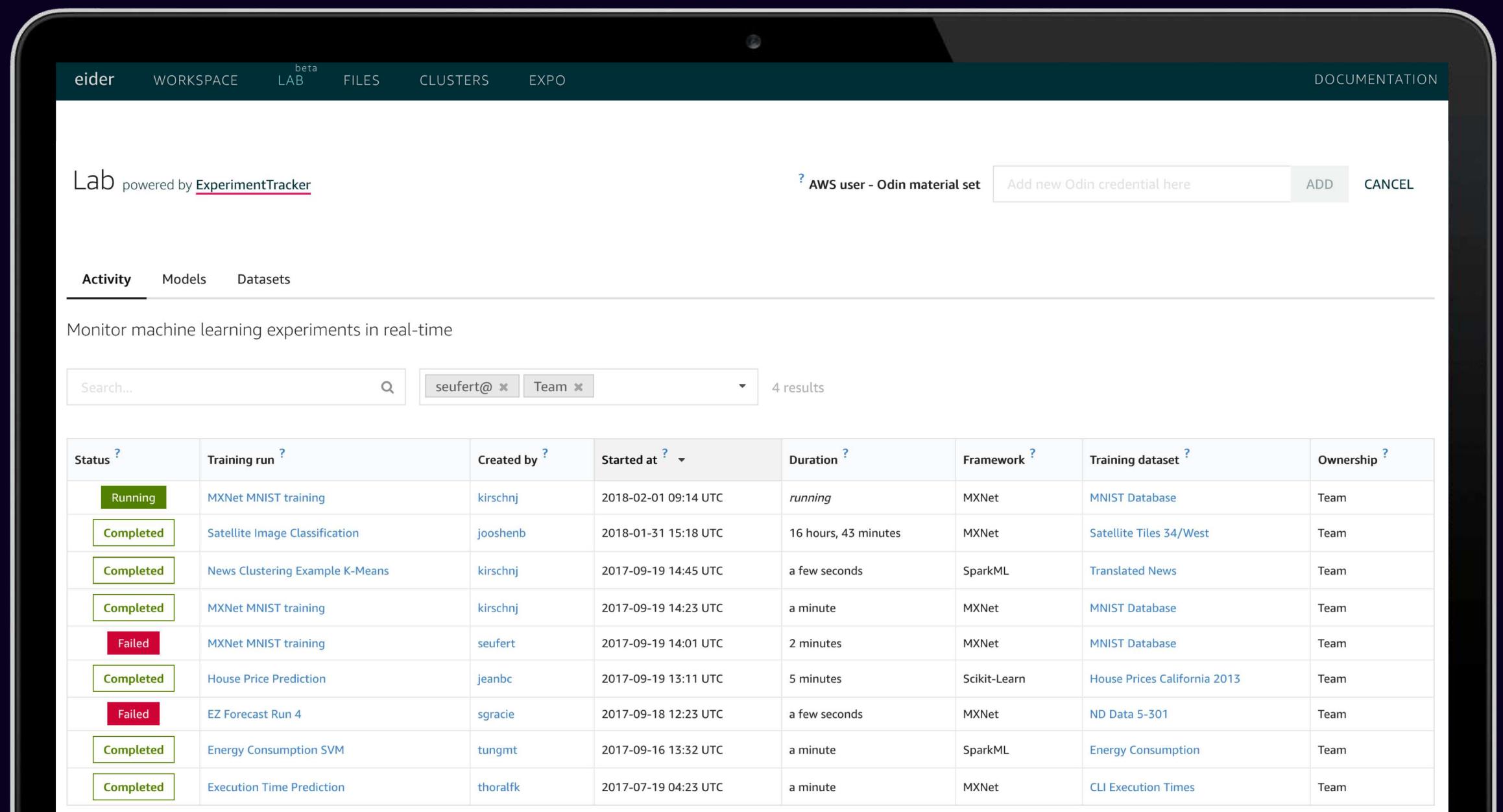
Automated Metadata Extraction



Data Access



Browse experiments



The screenshot shows the eider platform interface, specifically the 'Lab' section for monitoring machine learning experiments. The top navigation bar includes 'eider', 'WORKSPACE', 'LAB ^{beta}', 'FILES', 'CLUSTERS', 'EXPO', and 'DOCUMENTATION'. The 'LAB' tab is active.

The 'Lab' section is powered by [ExperimentTracker](#). A modal window titled 'AWS user - Odin material set' is open, with a text input field 'Add new Odin credential here', an 'ADD' button, and a 'CANCEL' button.

Below the modal, there are tabs for 'Activity', 'Models', and 'Datasets', with 'Activity' being the active tab. A sub-header says 'Monitor machine learning experiments in real-time'. A search bar shows 'Search...', a user filter for 'seufert@', a team filter for 'Team', and a dropdown showing '4 results'.

The main table displays 10 experiment entries:

Status ?	Training run ?	Created by ?	Started at ? ▾	Duration ?	Framework ?	Training dataset ?	Ownership ?
Running	MXNet MNIST training	kirschnj	2018-02-01 09:14 UTC	running	MXNet	MNIST Database	Team
Completed	Satellite Image Classification	jooshenb	2018-01-31 15:18 UTC	16 hours, 43 minutes	MXNet	Satellite Tiles 34/West	Team
Completed	News Clustering Example K-Means	kirschnj	2017-09-19 14:45 UTC	a few seconds	SparkML	Translated News	Team
Completed	MXNet MNIST training	kirschnj	2017-09-19 14:23 UTC	a minute	MXNet	MNIST Database	Team
Failed	MXNet MNIST training	seufert	2017-09-19 14:01 UTC	2 minutes	MXNet	MNIST Database	Team
Completed	House Price Prediction	jeanbc	2017-09-19 13:11 UTC	5 minutes	Scikit-Learn	House Prices California 2013	Team
Failed	EZ Forecast Run 4	sgracie	2017-09-18 12:23 UTC	a few seconds	MXNet	ND Data 5-301	Team
Completed	Energy Consumption SVM	tungmt	2017-09-16 13:32 UTC	a minute	SparkML	Energy Consumption	Team
Completed	Execution Time Prediction	thoralfk	2017-07-19 04:23 UTC	a minute	MXNet	CLI Execution Times	Team

Monitor model training

eider WORKSPACE beta LAB FILES CLUSTERS EXPO DOCUMENTATION

BACK

Training run: MXNet MNIST training

Training Run Info Training Progress **Source**

Train Accuracy

Train-accuracy

Epoch	Timestamp	Value
0	2017-09-19 14:23:59:993 UTC	0.92241

Inspect artifacts

eider WORKSPACE ^{beta} LAB FILES CLUSTERS EXPO DOCUMENTATION

BACK

Model: SMS Spam Classification (Logistic Regression)

Model Info Hyperparameters Pipeline **Evaluations** Source

```
graph TD; INPUT[INPUT] --> message[message]; message --> Tokenizer[Tokenizer]; Tokenizer --> words[words]; words --> StopWordsRemover[StopWordsRemover]; StopWordsRemover --> wordsWithoutStopWords[wordsWithoutStopWords]; labeling[labeling] --> StringIndexerModel[StringIndexerModel]; StringIndexerModel --> label[label]; wordsWithoutStopWords --> HashingTF[HashingTF]; HashingTF --> features[features]; label --> OUTPUT[OUTPUT]; features --> OUTPUT
```

The pipeline diagram illustrates the data flow for SMS Spam Classification. It starts with an **INPUT** node, which points to a **message** node. This is followed by a **Tokenizer** node, which leads to a **words** node. The **words** node then points to a **StopWordsRemover** node, which leads to a **wordsWithoutStopWords** node. The **wordsWithoutStopWords** node and a **labeling** node both point to a **StringIndexerModel** node, which in turn points to a **label** node. The **wordsWithoutStopWords** node also points to a **HashingTF** node, which points to a **features** node. Finally, the **label** and **features** nodes both point to an **OUTPUT** node.

Transform details

Id	89a07589-5cc6-4022-a109-3824e0100124
Name	HashingTF
Framework	

Transform parameters

Parameter	Value
numFeatures	1000
binary	false

1. Interactive Tracking

Eider Workspace Integration

The screenshot shows the Eider workspace interface. The top navigation bar includes 'eider' (selected), 'WORKSPACE' (selected), 'beta LAB', 'FILES', 'CLUSTERS', and 'EXPO'. Below the navigation is a secondary menu with 'Notebook' (selected), 'Cell', 'System', and 'Window'. The main content area displays a Scala notebook titled 'SMS Spam Classification using Scala, SparkML, and Ryck ExperimentTracker'. The code in the notebook is as follows:

```
1 import org.apache.spark.ml.{Pipeline, PipelineStage}
2 import org.apache.spark.ml.classification.LogisticRegression
3 import org.apache.spark.ml.feature.{HashingTF, StopWordsRemover, StringIndexer, Tokenizer}
4
5 val labelIndexer = new StringIndexer()
6   .setInputCol("labeling")
7   .setOutputCol("label")
8 val tokenizer = new Tokenizer()
9   .setInputCol("message")
10  .setOutputCol("words")
11 val stopWordsRemover = new StopWordsRemover()
12  .setInputCol(tokenizer.getOutputCol)
13  .setOutputCol("wordsWithoutStopWords")
14 val hashingTF = new HashingTF()
15  .setNumFeatures(1000)
16  .setInputCol(stopWordsRemover.getOutputCol)
17  .setOutputCol("features")
18
19 val learner = new LogisticRegression()
20  .setMaxIter(5)
21
22 val pipeline = new Pipeline()
23  .setStages(Array(labelIndexer, tokenizer, stopWordsRemover, hashingTF, learner))
```

The right sidebar contains a sidebar navigation with 'AWS account', 'Datasets', 'History', 'Models', 'Server', and a 'Variables' section. The 'Variables' section includes a search bar and a list of variables:

Variable	Value
username	seufert
username	seufert
awsAccessKeyId	AKIAJHH20EQDPXM
awsSecretKey	value hidden
awsAccessKeyId	AKIAJHH20EQDPXM
awsSecretKey	value hidden
sc	<spark.context>
spark	<spark.sql.session>
eider	<eider_python.eider>
awsAccessKeyId	AKIAJHH20EQDPXM
awsSecretKey	value hidden
username	seufert
spark	Java ref type of
sc	Java ref type of
trackerApi	ExperimentTrack
eider	com.amazon.eide

Eider: Interactive Tracking

New design – coming soon

```
15  .setNumFeatures(1000)
16  .setInputCol(stopWordsRemover.getOutputCol)
17  .setOutputCol("features")
18
19  val learner = new LogisticRegression()
20  .setMaxIter(5)
21
22  val pipeline = new Pipeline()
23  .setStages(Array(labelIndexer, tokenizer, stopWordsRemover, hashingTF, learner))
24
25  val pipelineModel = pipeline.fit(trainingDataset)
```

▶ RUN | Scala | Last run: 1.5 s

● pipelineModel

Untracked

TRACK IN LAB WINDOW

...

Eider: Interactive Tracking

New design – coming soon

```
15  .setNumFeatures(1000)
16  .setInputCol(stopWordsRemover.getOutputCol)
17  .setOutputCol("features")
18
19  val learner = new LogisticRegression()
20  .setMaxIter(5)
21
22  val pipeline = new Pipeline()
23  .setStages(Array(labelIndexer, tokenizer, stopWordsRemover, hashingTF, learner))
24
25  val pipelineModel = pipeline.fit(trainingDataset)
```

▶ RUN | Scala | Last run: 1.5 s

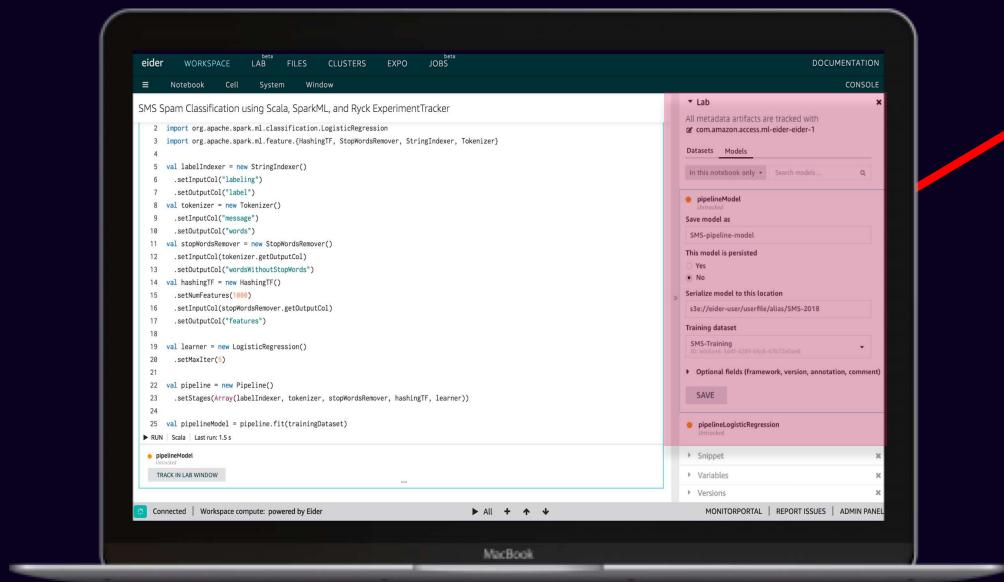
● pipelineModel

Untracked

TRACK IN LAB WINDOW

Eider: Interactive Tracking

New design – coming soon



▼ Lab

All metadata artifacts are tracked with com.amazon.access.ml-eider-eider-1

Datasets Models

In this notebook only Search models...

pipelineModel Untracked

Save model as SMS-pipeline-model

This model is persisted

Yes No

Serialize model to this location s3://eider-user/userfile/alias/SMS-2018

Training dataset

SMS-Training

ID: b0c6ae6-3d4f-4289-b6cb-67b72e0ae6

▶ Optional fields (framework, version, annotation, comment)

SAVE

2. Log Analysis

Logexp: Non-Intrusive Experiment Tracking

```
./run-mxnet-experiment.py
```

Logexp: Non-Intrusive Experiment Tracking

```
./run-mxnet-experiment.py
```

```
INFO:root:Epoch[0] Batch [100] Speed: 83831.42 samples/sec accuracy=0.10432
INFO:root:Epoch[0] Batch [200] Speed: 59385.85 samples/sec accuracy=0.11060
INFO:root:Epoch[0] Batch [300] Speed: 62468.41 samples/sec accuracy=0.11310
INFO:root:Epoch[0] Batch [400] Speed: 68188.53 samples/sec accuracy=0.11480
INFO:root:Epoch[0] Batch [500] Speed: 40556.61 samples/sec accuracy=0.12300
INFO:root:Epoch[0] Train-accuracy=0.300101
INFO:root:Epoch[0] Time cost=1.210
INFO:root:Epoch[0] Validation-accuracy=0.457100
INFO:root:Epoch[1] Batch [100] Speed: 38430.77 samples/sec accuracy=0.547426
. . .
```

Logexp: Non-Intrusive Experiment Tracking

```
> pip install logexp
./run-mxnet-experiment.py

INFO:root:Epoch[0] Batch [100] Speed: 83831.42 samples/sec accuracy=0.10432
INFO:root:Epoch[0] Batch [200] Speed: 59385.85 samples/sec accuracy=0.11060
INFO:root:Epoch[0] Batch [300] Speed: 62468.41 samples/sec accuracy=0.11310
INFO:root:Epoch[0] Batch [400] Speed: 68188.53 samples/sec accuracy=0.11480
INFO:root:Epoch[0] Batch [500] Speed: 40556.61 samples/sec accuracy=0.12300
INFO:root:Epoch[0] Train-accuracy=0.300101
INFO:root:Epoch[0] Time cost=1.210
INFO:root:Epoch[0] Validation-accuracy=0.457100
INFO:root:Epoch[1] Batch [100] Speed: 38430.77 samples/sec accuracy=0.547426
. . .
```

Logexp: Non-Intrusive Experiment Tracking

```
> pip install logexp
> logexp-run ./run-mxnet-experiment.py

INFO:root:Epoch[0] Batch [100] Speed: 83831.42 samples/sec accuracy=0.10432
INFO:root:Epoch[0] Batch [200] Speed: 59385.85 samples/sec accuracy=0.11060
INFO:root:Epoch[0] Batch [300] Speed: 62468.41 samples/sec accuracy=0.11310
INFO:root:Epoch[0] Batch [400] Speed: 68188.53 samples/sec accuracy=0.11480
INFO:root:Epoch[0] Batch [500] Speed: 40556.61 samples/sec accuracy=0.12300
INFO:root:Epoch[0] Train-accuracy=0.300101
INFO:root:Epoch[0] Time cost=1.210
INFO:root:Epoch[0] Validation-accuracy=0.457100
INFO:root:Epoch[1] Batch [100] Speed: 38430.77 samples/sec accuracy=0.547426
. . .
```

Logexp: Non-Intrusive Experiment Tracking

```
> pip install logexp  
> logexp-run ./run-mxnet-experiment.py  
  
INFO:root:Epoch[0] Batch [100] Speed: 83831.42 samples/sec accuracy=0.10432  
INFO:root:Epoch[0] Batch [200] Speed: 59385.85 samples/sec accuracy=0.11060  
INFO:root:Epoch[0] Batch [300] Speed: 62468.41 samples/sec accuracy=0.11310  
INFO:root:Epoch[0] Batch [400] Speed: 68188.53 samples/sec accuracy=0.11480  
INFO:root:Epoch[0] Batch [500] Speed: 40556.61 samples/sec accuracy=0.12300  
INFO:root:Epoch[0] Train-accuracy=0.300101  
INFO:root:Epoch[0] Time cost=1.210  
INFO:root:Epoch[0] Validation-accuracy=0.457100  
INFO:root:Epoch[1] Batch [100] Speed: 38430.77 samples/sec accuracy=0.547426  
. . .
```

Information Extraction

Logexp: Non-Intrusive Experiment Tracking

```
> pip install logexp  
> logexp-run ./run-mxnet-experiment.py
```

```
INFO:root:Epoch[0] Batch [100] Speed: 83831.42 samples/sec accuracy=0.10432  
INFO:root:Epoch[0] Batch [200] Speed: 59385.85 samples/sec accuracy=0.11060  
INFO:root:Epoch[0] Batch [300] Speed: 62468.41 samples/sec accuracy=0.11310  
INFO:root:Epoch[0] Batch [400] Speed: 68188.53 samples/sec accuracy=0.11480  
INFO:root:Epoch[0] Batch [500] Speed: 40556.61 samples/sec accuracy=0.12300  
INFO:root:Epoch[0] Train-accuracy=0.300101  
INFO:root:Epoch[0] Time cost=1.210  
INFO:root:Epoch[0] Validation-accuracy=0.457100  
INFO:root:Epoch[1] Batch [100] Speed: 38430.77 samples/sec accuracy=0.547426  
. . .
```

Environment Information

Information Extraction

Logexp: Non-Intrusive Experiment Tracking

```
> pip install logexp  
> logexp-run ./run-mxnet-experiment.py
```

Environment Information

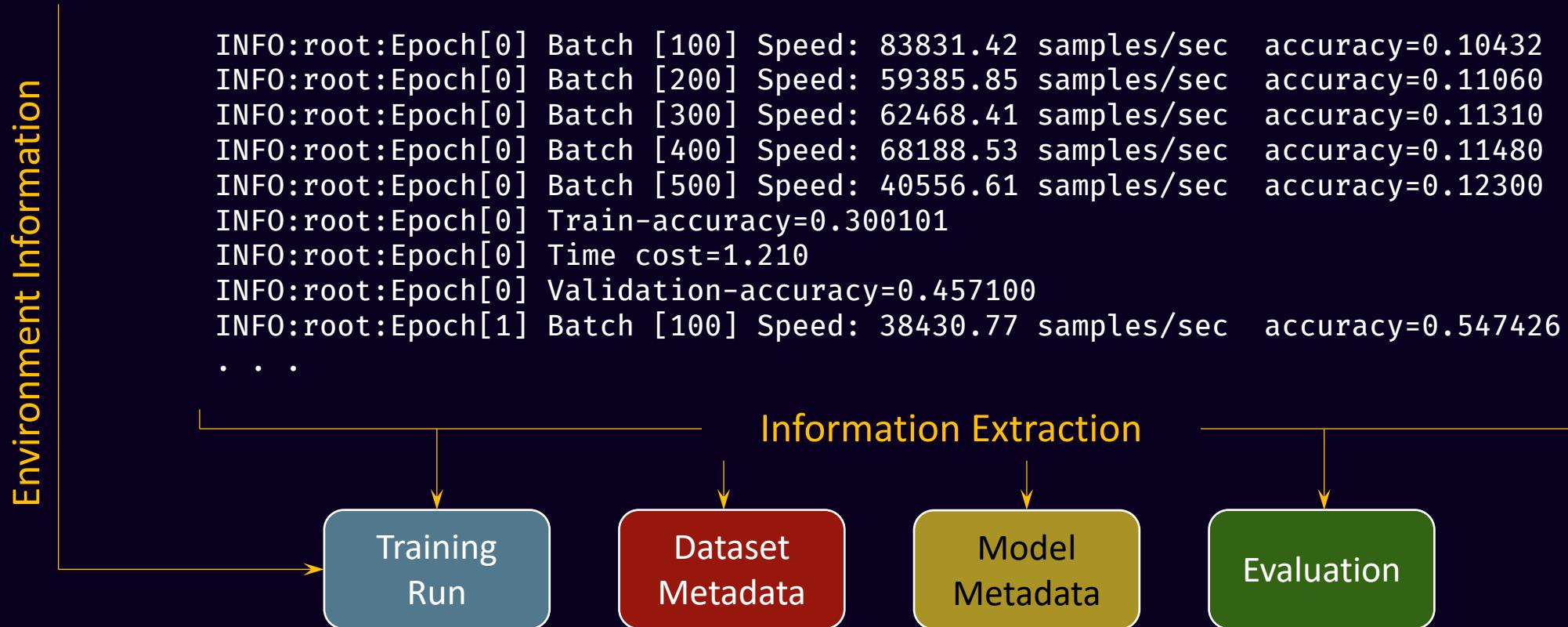
```
INFO:root:Epoch[0] Batch [100] Speed: 83831.42 samples/sec accuracy=0.10432  
INFO:root:Epoch[0] Batch [200] Speed: 59385.85 samples/sec accuracy=0.11060  
INFO:root:Epoch[0] Batch [300] Speed: 62468.41 samples/sec accuracy=0.11310  
INFO:root:Epoch[0] Batch [400] Speed: 68188.53 samples/sec accuracy=0.11480  
INFO:root:Epoch[0] Batch [500] Speed: 40556.61 samples/sec accuracy=0.12300  
INFO:root:Epoch[0] Train-accuracy=0.300101  
INFO:root:Epoch[0] Time cost=1.210  
INFO:root:Epoch[0] Validation-accuracy=0.457100  
INFO:root:Epoch[1] Batch [100] Speed: 38430.77 samples/sec accuracy=0.547426  
. . .
```

Information Extraction

Logexp: Non-Intrusive Experiment Tracking

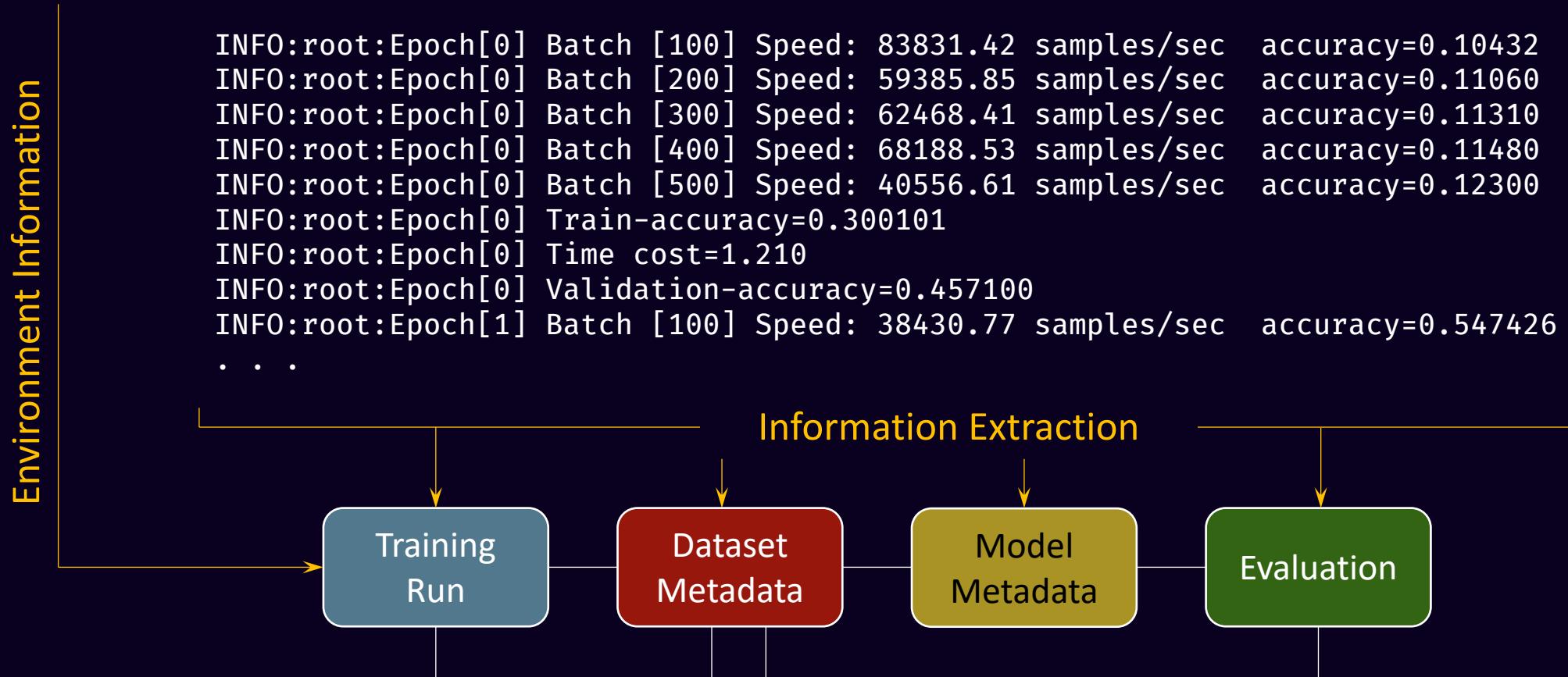
```
> pip install logexp
```

```
> logexp-run ./run-mxnet-experiment.py
```



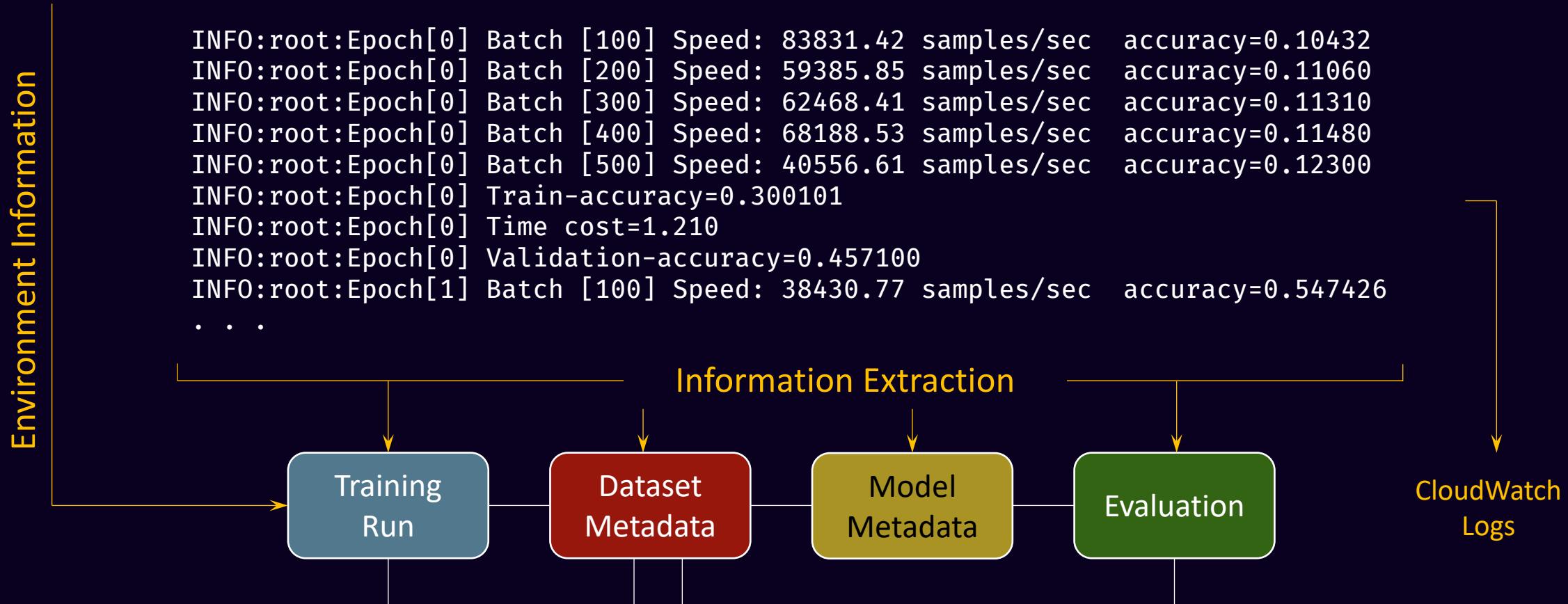
Logexp: Non-Intrusive Experiment Tracking

```
> pip install logexp  
> logexp-run ./run-mxnet-experiment.py
```



Logexp: Non-Intrusive Experiment Tracking

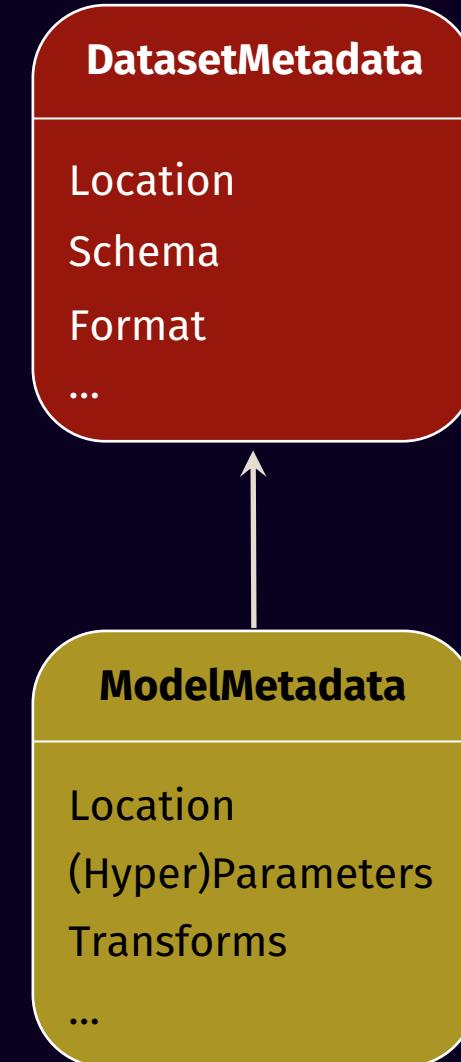
```
> pip install logexp  
> logexp-run ./run-mxnet-experiment.py
```



3. Transparent Tracking

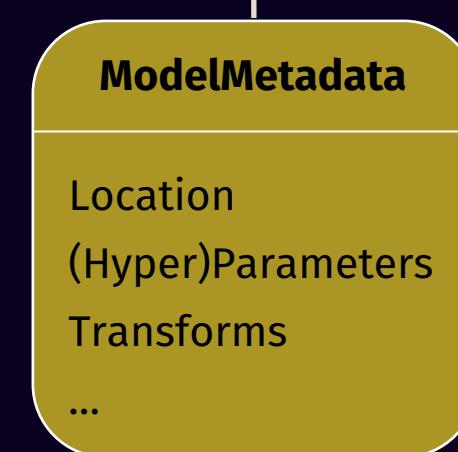
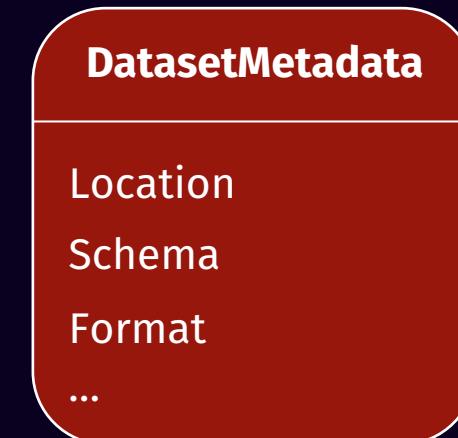
Automated Metadata Extraction (Recap)

```
val df: DataFrame = ...  
tracker.trackDataset(df)  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)  
tracker.trackModel(model)
```



Automated Metadata Extraction (Recap)

```
val df: DataFrame = ...  
tracker.trackDataset(df)  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)  
tracker.trackModel(model)
```



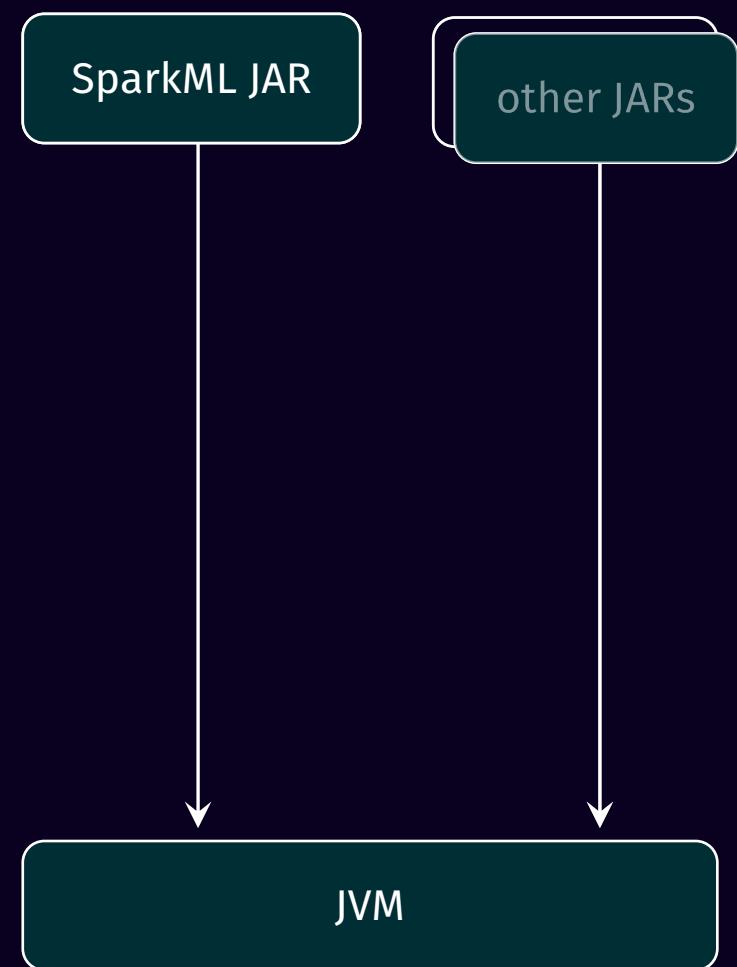
Transparent Tracking - JVM

Transparent Tracking - JVM

```
> java -cp sparkml.jar:... . . .
```

Transparent Tracking - JVM

```
> java -cp sparkml.jar:... . . .
```



Transparent Tracking - JVM

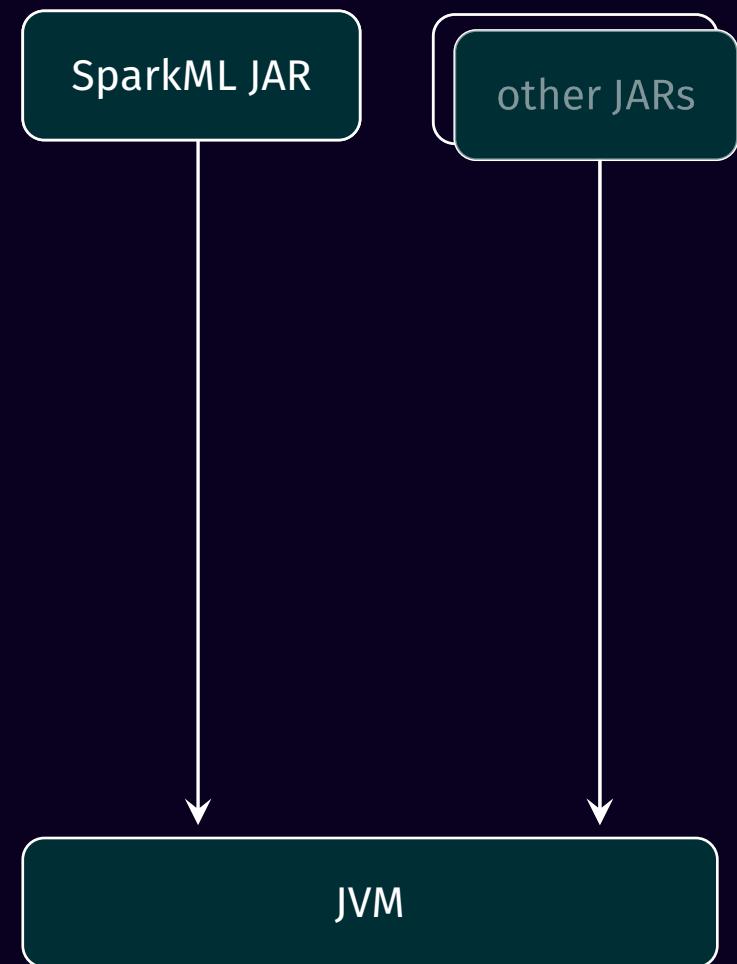
```
> java -cp sparkml.jar:... . . .
```

```
import org.apache.spark.ml.Pipeline

val df: DataFrame = ...

val pipeline: Pipeline = ...

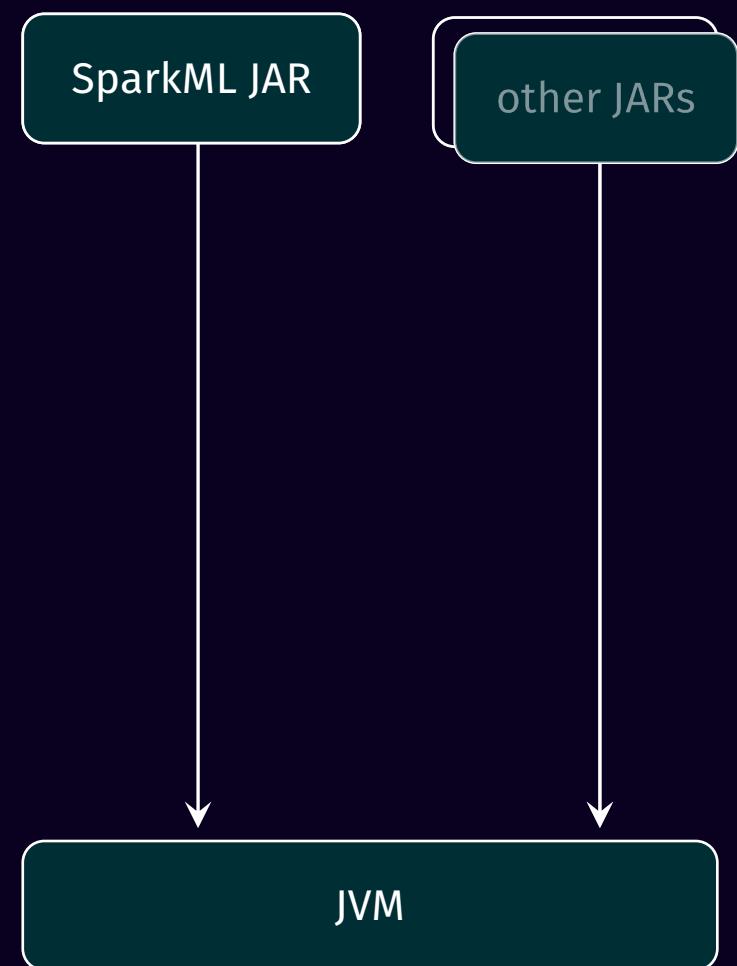
val model: PipelineModel = pipeline.fit(df)
```



Transparent Tracking - JVM

```
> java -javaagent:tracker.jar -cp sparkml.jar:... . . .
```

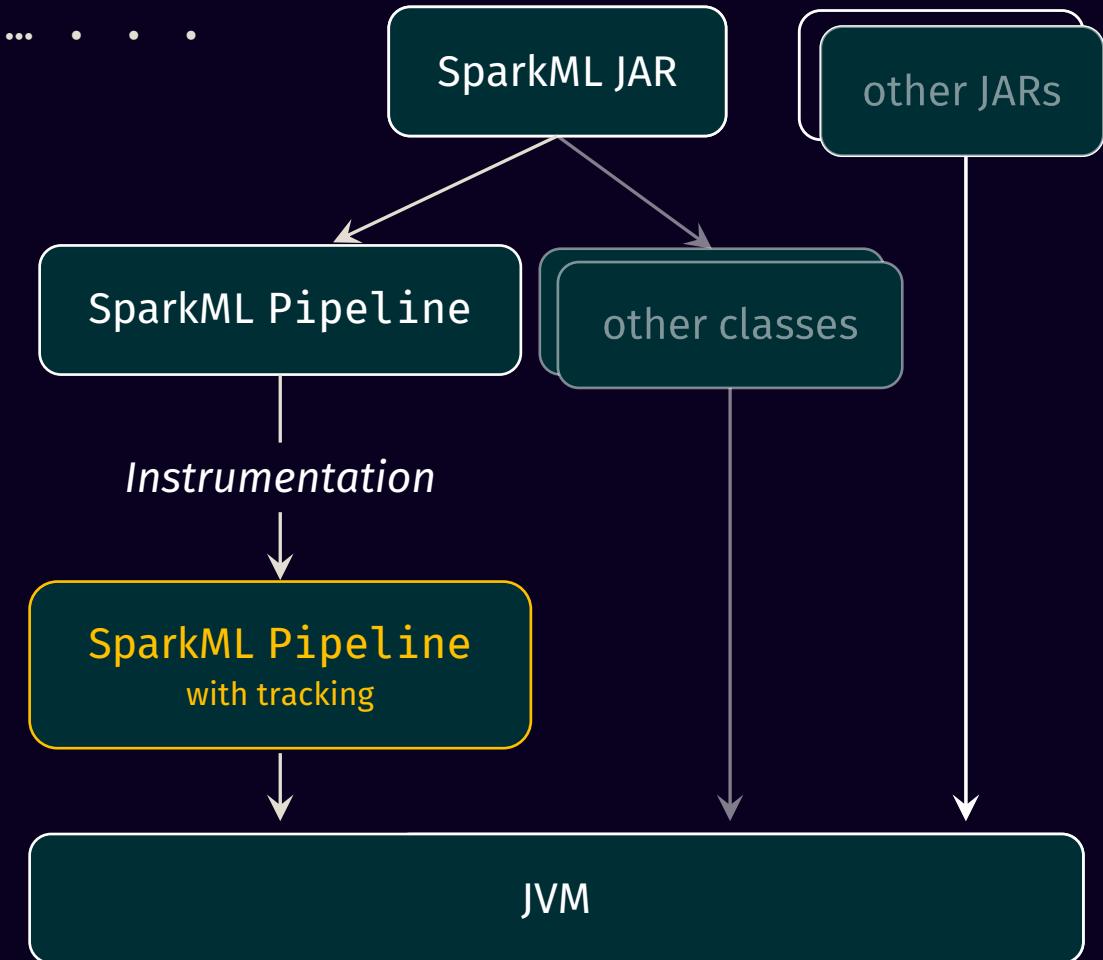
```
import org.apache.spark.ml.Pipeline  
  
val df: DataFrame = ...  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)
```



Transparent Tracking - JVM

```
> java -javaagent:tracker.jar -cp sparkml.jar:... . . .
```

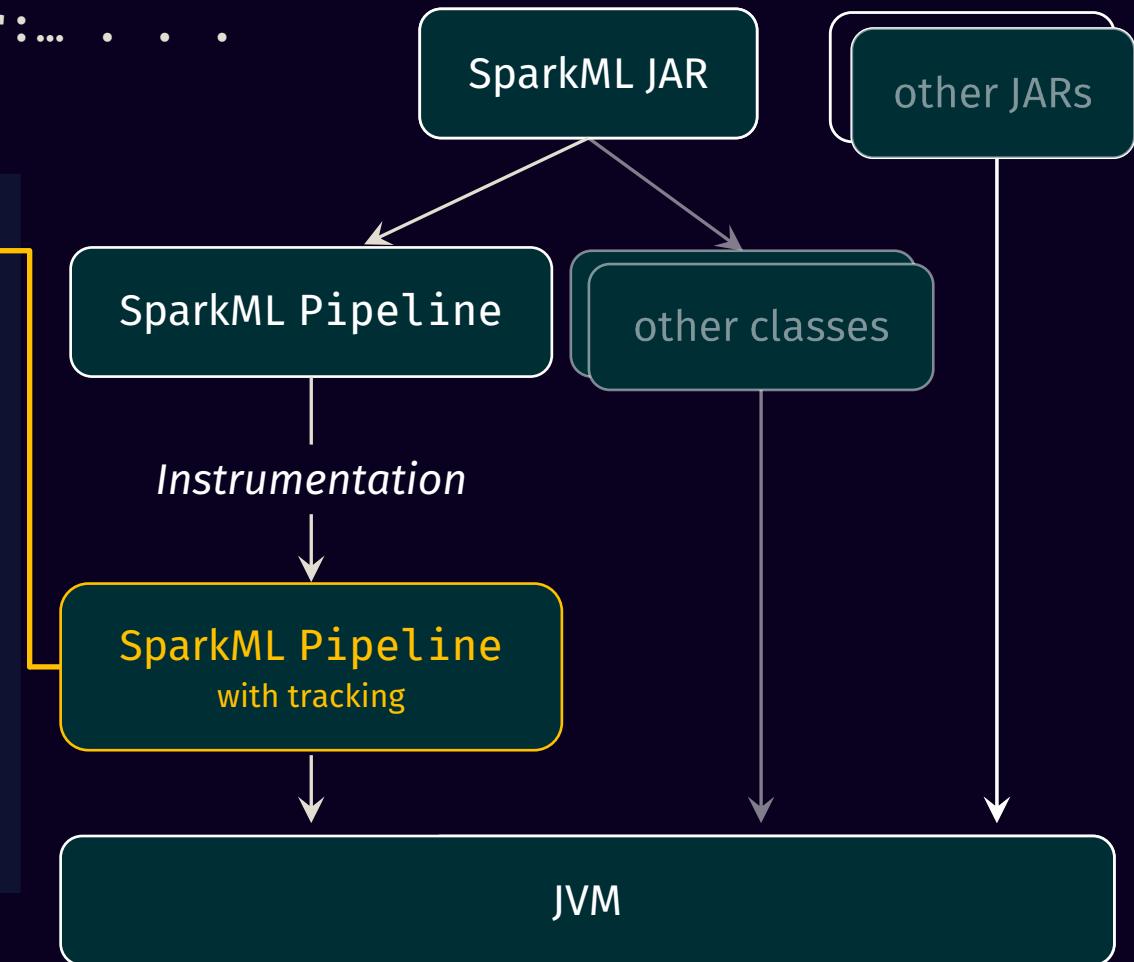
```
import org.apache.spark.ml.Pipeline  
  
val df: DataFrame = ...  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)
```



Transparent Tracking - JVM

```
> java -javaagent:tracker.jar -cp sparkml.jar:... . . .
```

```
import org.apache.spark.ml.Pipeline  
  
val df: DataFrame = ...  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)
```



Transparent Tracking - JVM

```
> java -javaagent:tracker.jar -cp sparkml.jar:... . . .
```

```
import org.apache.spark.ml.Pipeline

val df: DataFrame = ...

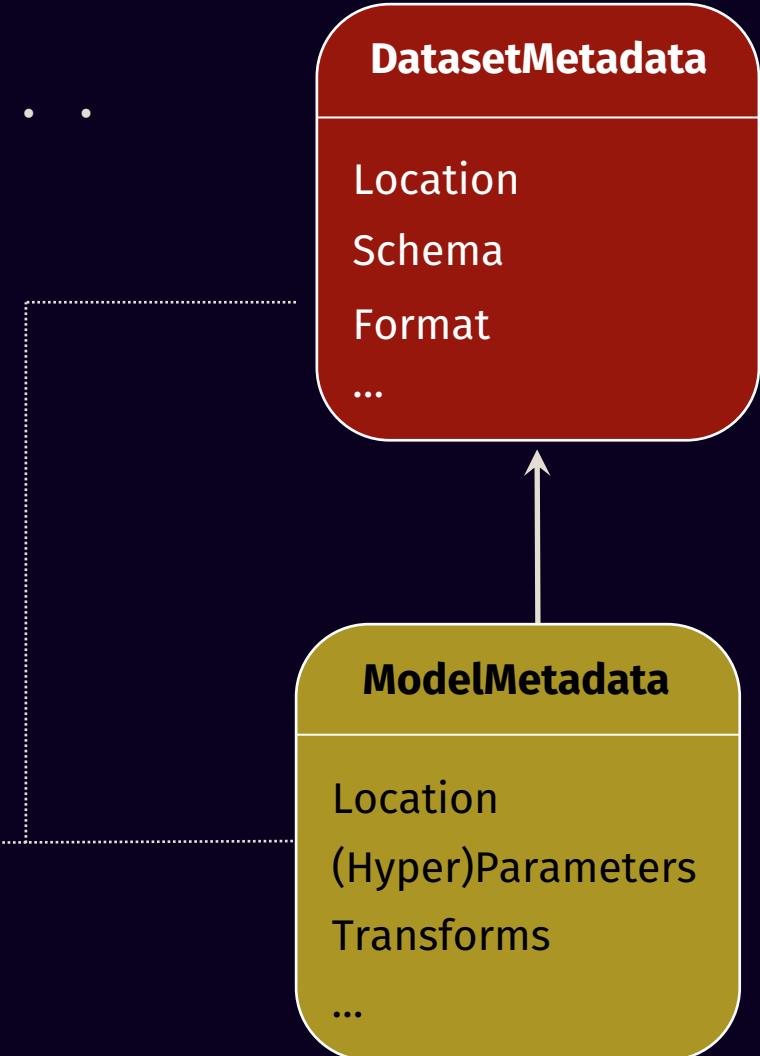
val pipeline: Pipeline = ...

val model: PipelineModel = pipeline.fit(df)
```

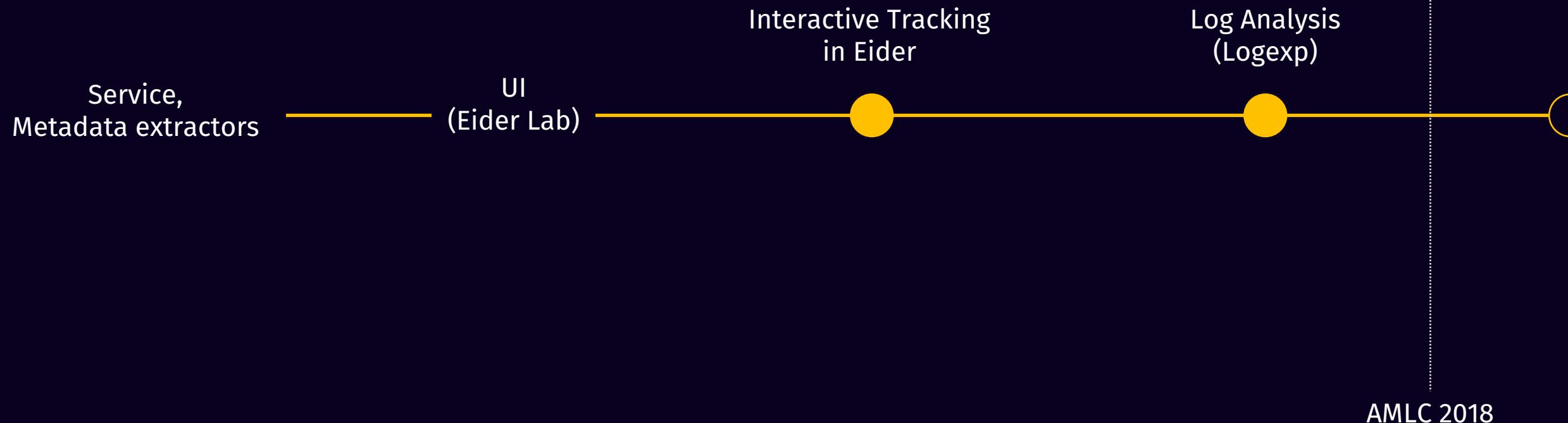
Transparent Tracking - JVM

```
> java -javaagent:tracker.jar -cp sparkml.jar:... . . .
```

```
import org.apache.spark.ml.Pipeline  
  
val df: DataFrame = ...  
  
val pipeline: Pipeline = ...  
  
val model: PipelineModel = pipeline.fit(df)
```

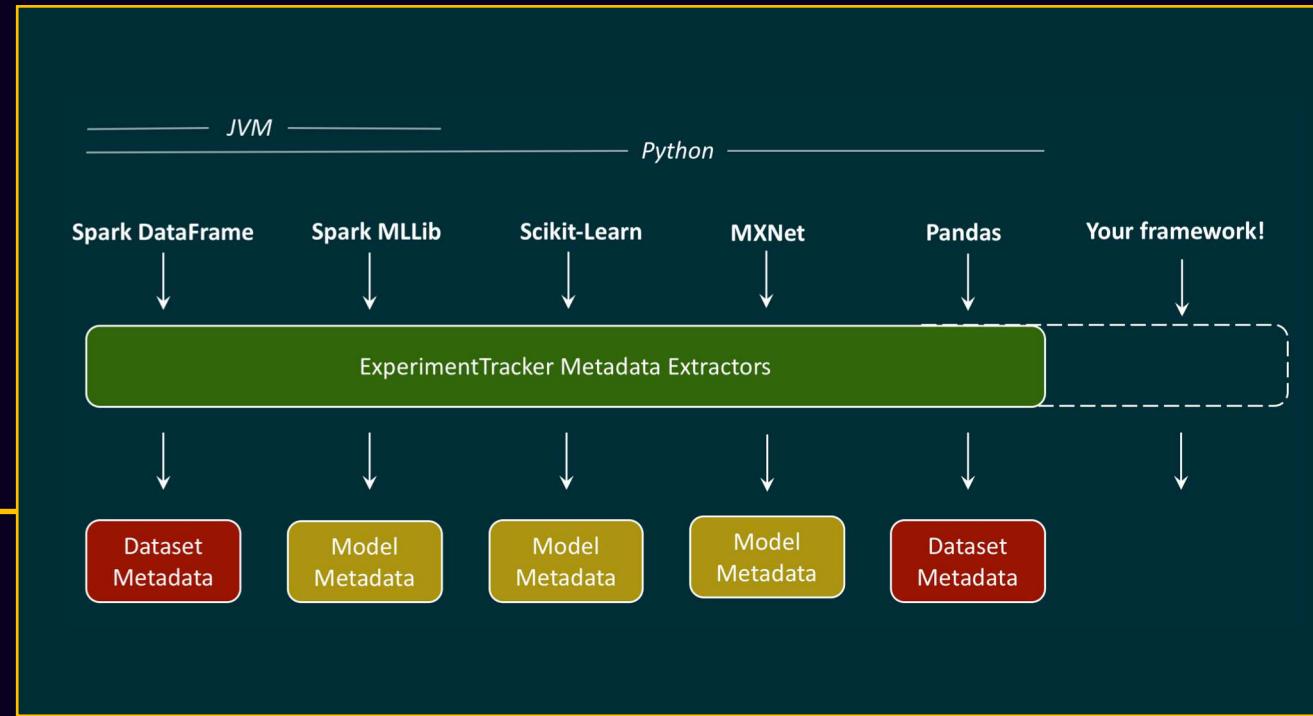


Summary & Timeline



Summary & Timeline

Service,
Metadata extractors

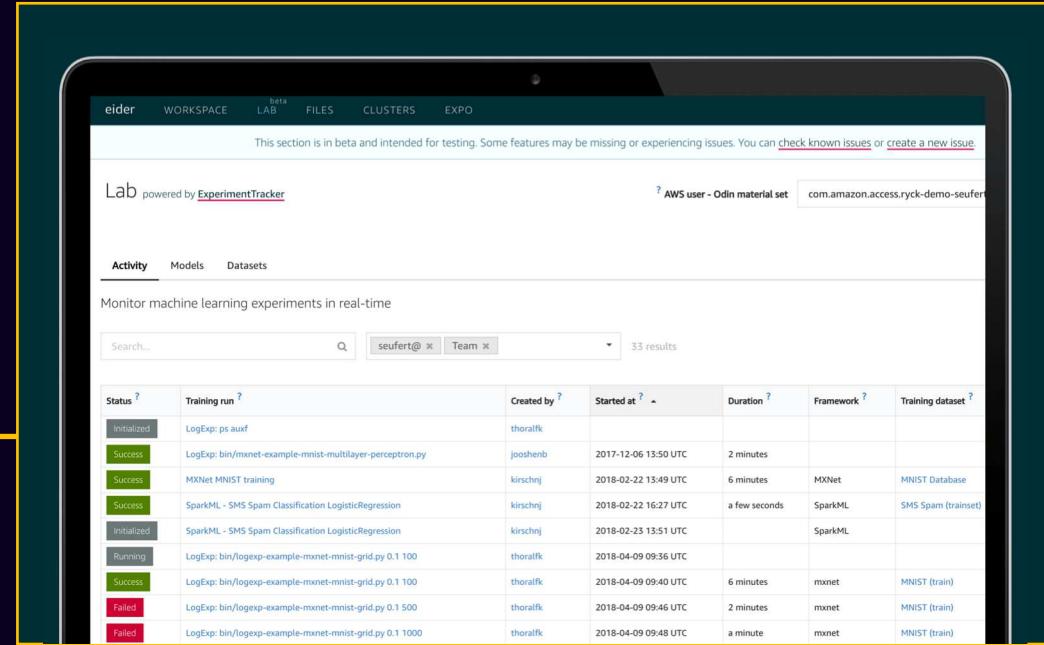


AMLC 2018

Summary & Timeline

Service,
Metadata extractors

UI
(Eider Lab)



AMLC 2018

Summary & Timeline

```
15 .setNumFeatures(1000)
16 .setInputCol(stopWordsRemover.getOutputCol)
17 .setOutputCol("features")
18
19 val learner = new LogisticRegression()
20 .setMaxIter(5)
21
22 val pipeline = new Pipeline()
23 .setStages(Array(labelIndexer, tokenizer, stopWordsRemover, hashingTF, learner))
24
25 val pipelineModel = pipeline.fit(trainingDataset)
```

▶ RUN | Scala | Last run: 1.5 s

● pipelineModel
Untracked

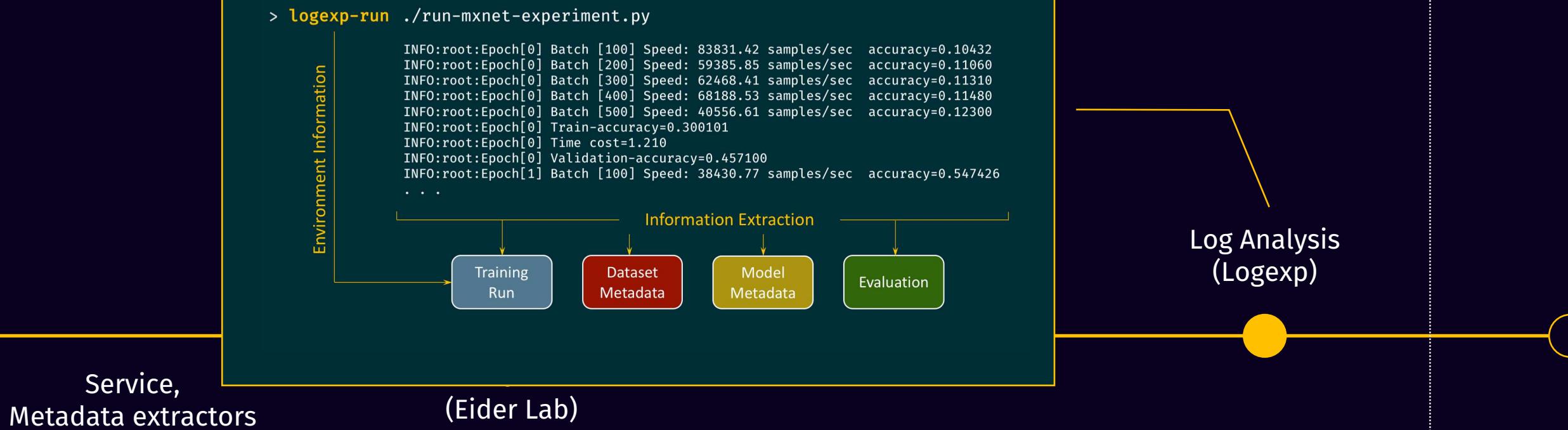
TRACK IN LAB WINDOW

Interactive Tracking in Eider

Log Analysis (Logexp)

AMLC 2018

Summary 8: Timeline



AMLC 2018

Outlook

Transparent Tracking
(JVM, Python)

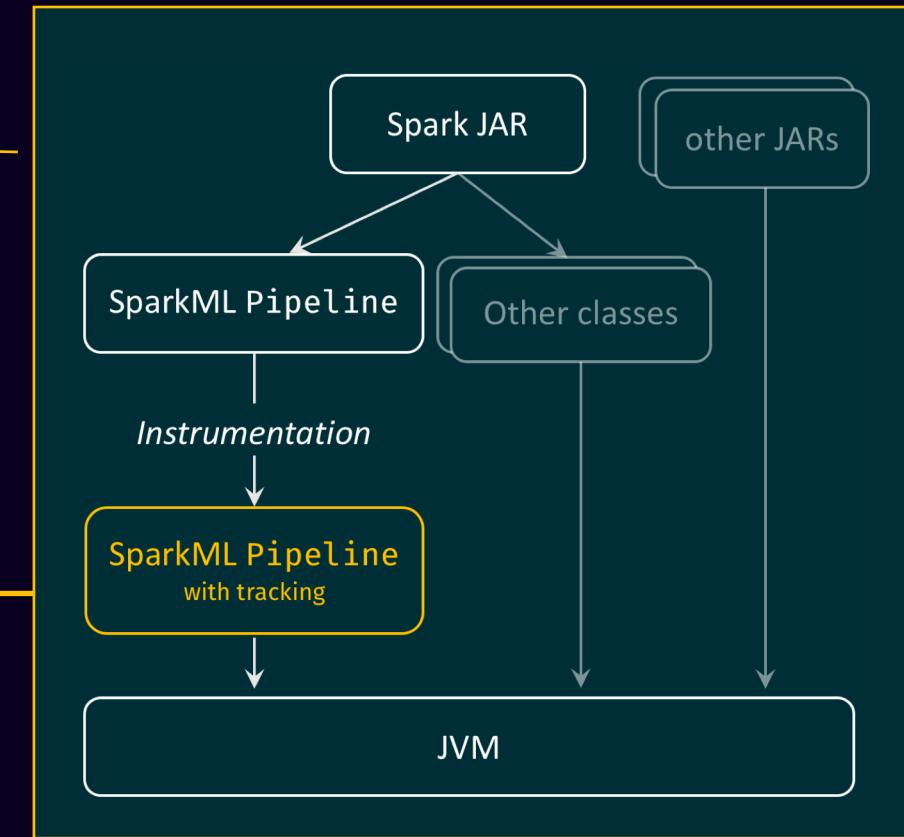
Productionization
Support



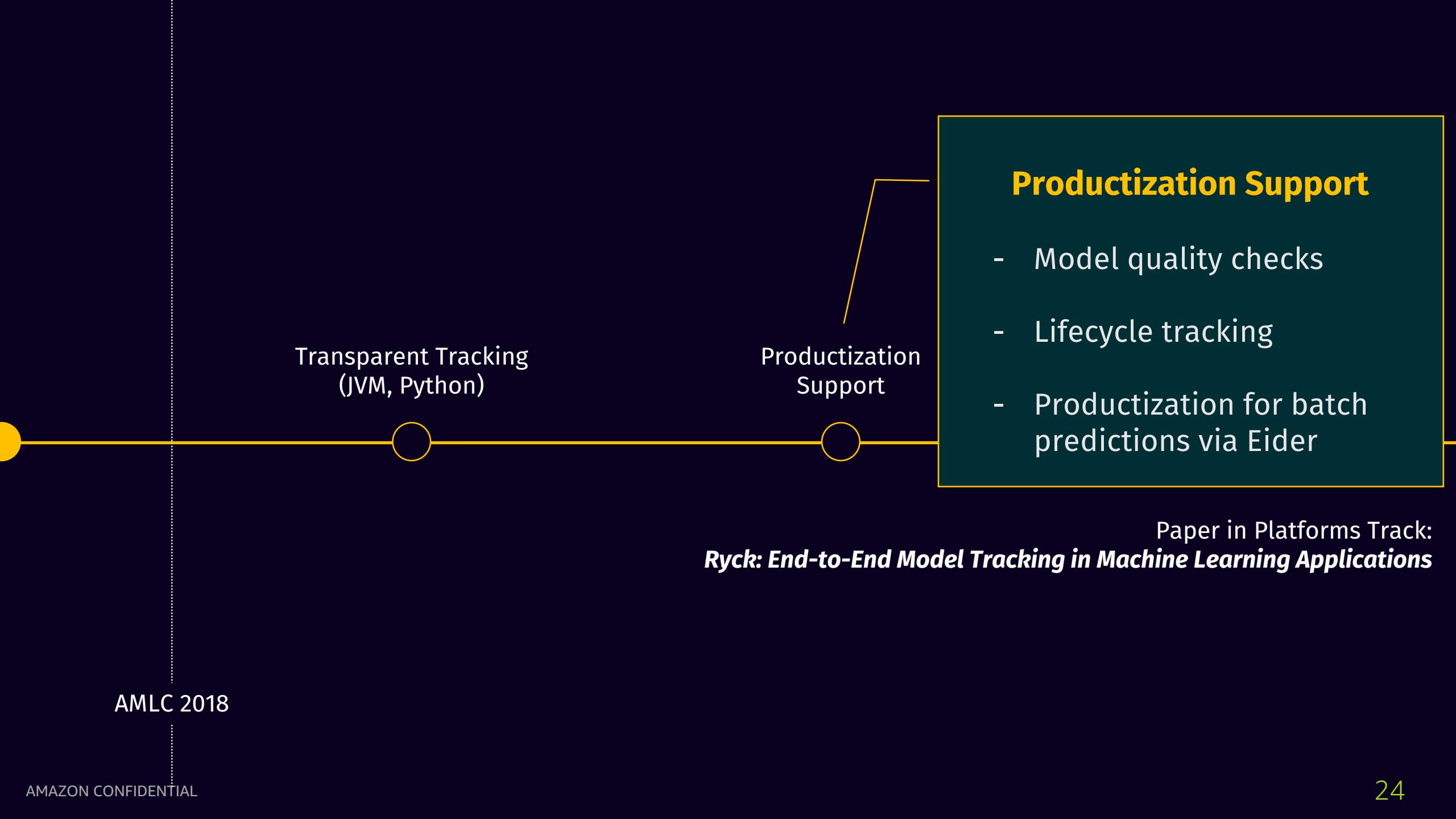
AMLC 2018

Outlook

Transparent Tracking
(JVM, Python)



AMLC 2018



Team

ExperimentTracker (Ryck)



Joos-Hendrik Boese
Manager AS
Berlin



Johannes Kirschnick
SDE
Berlin



Thoralf Klein
SDE
Berlin



Stephan Seufert
SDE
Berlin



Gabriel Zimmerman
Sr. SDE
New York City

Eider



Sam Gracie
Sr. UX Designer
Seattle



Margaret Tung
UX Designer
Seattle

40+
teams

160,000+
entities

eider.corp.amazon.com/lab

eider WORKSPACE LAB ^{beta} FILES CLUSTERS EXPO DOCUMENTATION

Get real-time status of your machine learning experiments

The Lab is powered by [ExperimentTracker](#), an easy-to-use service for extracting, storing, and maintaining metadata of machine learning artifacts of your team.

To get started, add your Odin credential.

Add new Odin credential here ADD CANCEL

CONTINUE



Monitor model training
Check the status of your model training runs in the activity tab.

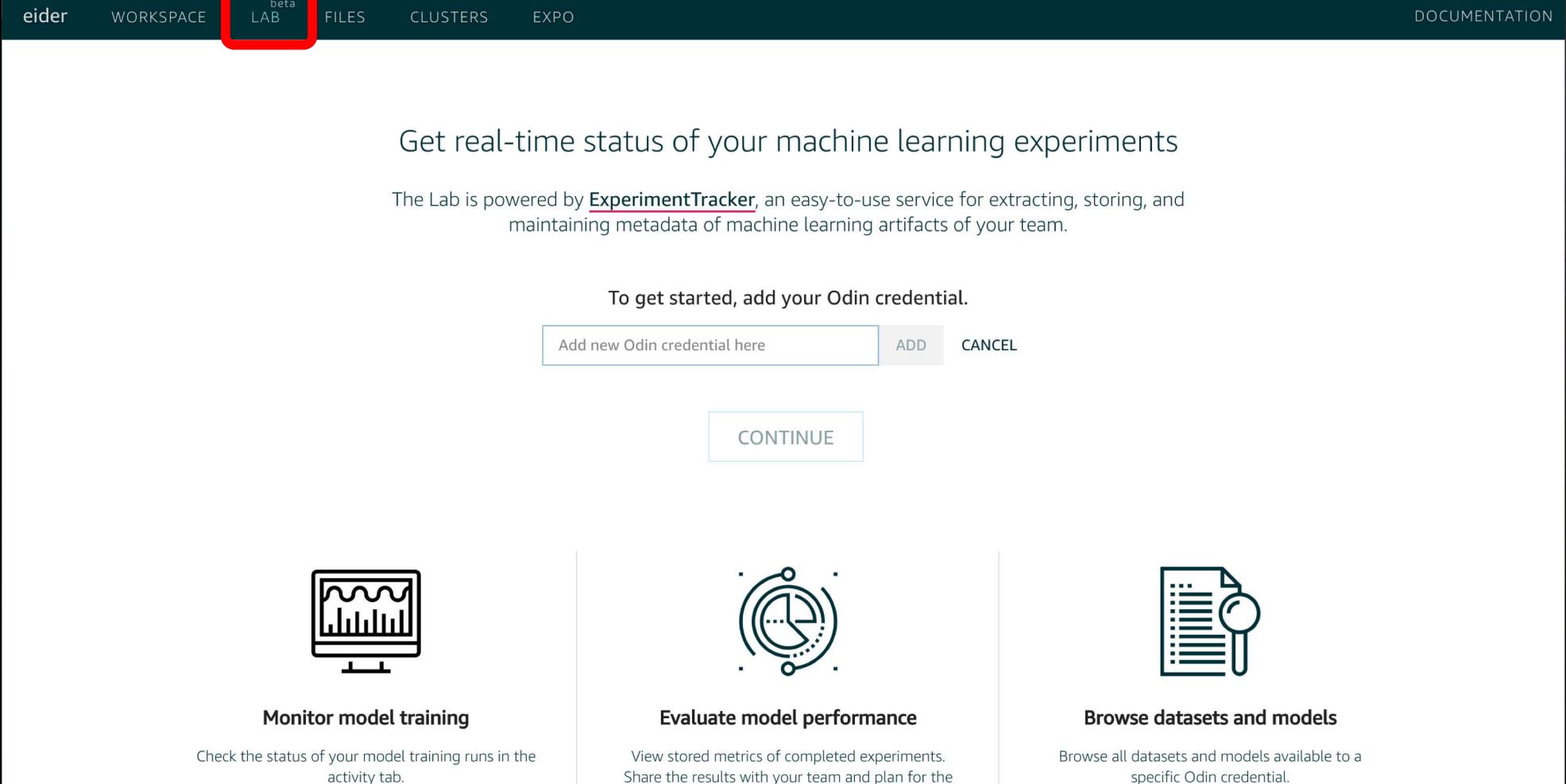


Evaluate model performance
View stored metrics of completed experiments. Share the results with your team and plan for the next steps.



Browse datasets and models
Browse all datasets and models available to a specific Odin credential.

eider.corp.amazon.com/lab



The screenshot shows the eider.corp.amazon.com/lab interface. At the top, there is a navigation bar with the following items: eider, WORKSPACE, LAB (which is highlighted with a red box), FILES, CLUSTERS, EXPO, and DOCUMENTATION. Below the navigation bar, the main content area has a heading "Get real-time status of your machine learning experiments". It explains that the Lab is powered by [ExperimentTracker](#), an easy-to-use service for extracting, storing, and maintaining metadata of machine learning artifacts of your team. A sub-section below this asks "To get started, add your Odin credential." with a "CONTINUE" button. At the bottom, there are three cards: "Monitor model training" (with a line chart icon), "Evaluate model performance" (with a pie chart icon), and "Browse datasets and models" (with a document and magnifying glass icon).

Get real-time status of your machine learning experiments

The Lab is powered by [ExperimentTracker](#), an easy-to-use service for extracting, storing, and maintaining metadata of machine learning artifacts of your team.

To get started, add your Odin credential.

Add new Odin credential here ADD CANCEL

CONTINUE



Monitor model training

Check the status of your model training runs in the activity tab.



Evaluate model performance

View stored metrics of completed experiments. Share the results with your team and plan for the next steps.



Browse datasets and models

Browse all datasets and models available to a specific Odin credential.

eider.corp.amazon.com/lab

eider WORKSPACE beta LAB FILES CLUSTERS EXPO DOCUMENTATION

Get real-time status of your machine learning experiments

The Lab is powered by [ExperimentTracker](#), an easy-to-use service for extracting, storing, and maintaining metadata of machine learning artifacts of your team.

To get started, add your Odin credential.

Add new Odin credential here ADD CANCEL

CONTINUE



Monitor model training

Check the status of your model training runs in the activity tab.



Evaluate model performance

View stored metrics of completed experiments. Share the results with your team and plan for the next steps.



Browse datasets and models

Browse all datasets and models available to a specific Odin credential.

eider.corp.amazon.com/lab

eider WORKSPACE LAB ^{beta} FILES CLUSTERS EXPO DOCUMENTATION

Get real-time status of your machine learning experiments

The Lab is powered by [ExperimentTracker](#), an easy-to-use service for extracting, storing, and maintaining metadata of machine learning artifacts of your team.

To get started, add your Odin credential.

Add new Odin credential here ADD CANCEL

CONTINUE



Monitor model training
Check the status of your model training runs in the activity tab.



Evaluate model performance
View stored metrics of completed experiments. Share the results with your team and plan for the next steps.



Browse datasets and models
Browse all datasets and models available to a specific Odin credential.