goto; copenhagen

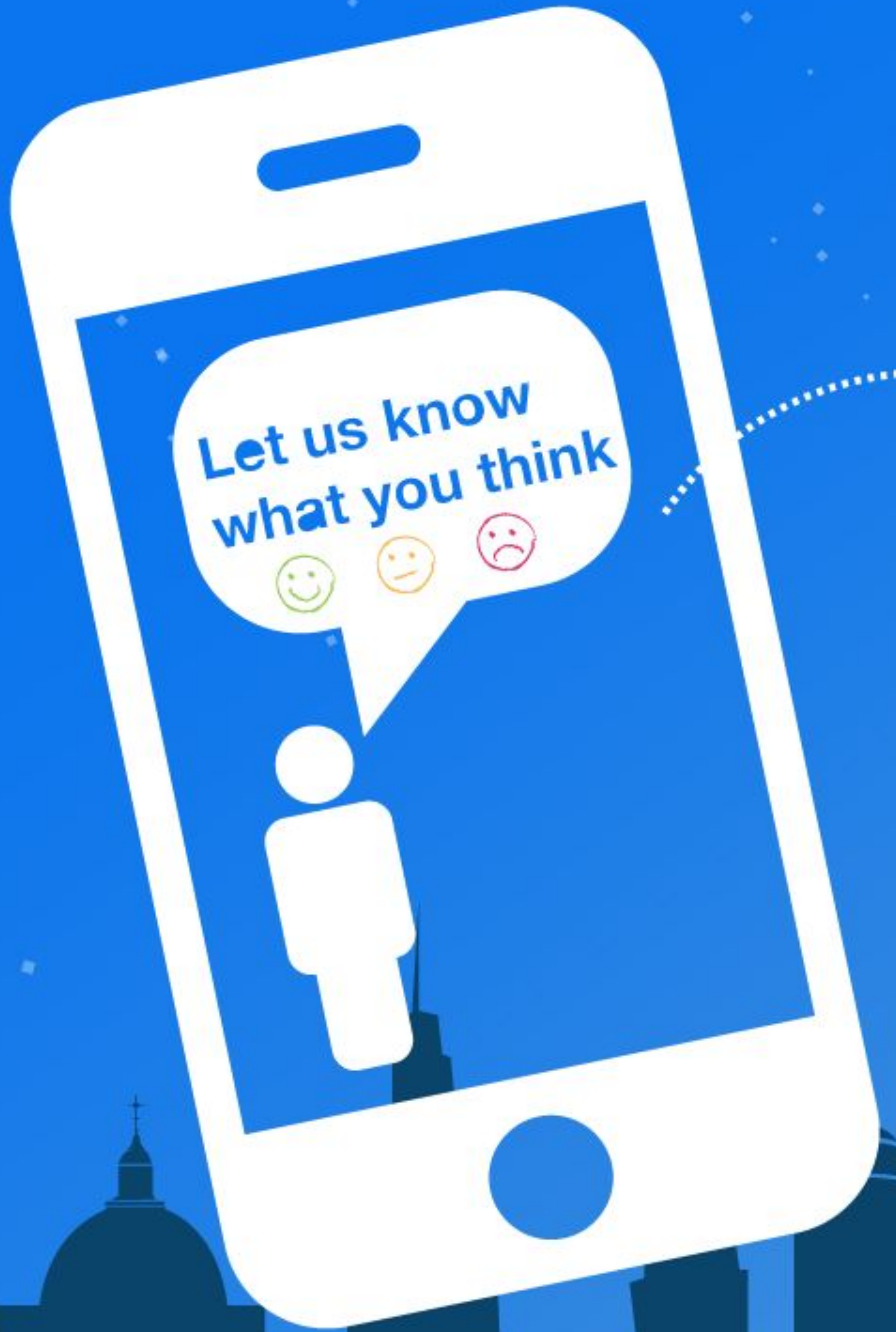
GOTO Copenhagen 2019
Conference Nov. 18 - 20

Taking Machine Learning Research to Production:
Solving Real Problems

Robert Crowe
Google

goto; copenhagen

GOTO Copenhagen 2019
Conference Nov. 18 - 20



Click 'Rate Session' to rate session and ask questions.

Follow us @gotocph

Imagine if you will ...

You're an Online Retailer Selling Shoes ...

Your model predicts **click-through rates (CTR)**, helping you decide how much inventory to order

When suddenly

Your AUC and prediction accuracy have **dropped** on men's dress shoes!

What causes problems?

Kinds of problems

- Fast - Example: bad sensor, bad software update
- Slow - Example: drift

Sudden Problems

Problem with data collection

- Bad sensor/camera
- Bad log data
- Moved or disabled sensors/cameras

Systems problem

- Bad software update
- Loss of network connectivity
- System down
- Bad credentials

Gradual Problems

Data changes

- Trend and seasonality
- Distribution of features changes
- Relative importance of features changes

World changes

- Styles change
- Competitors change
- Business expands to other geos

Why “Understand” the model?

Mispredictions do not have uniform **cost** to your business.

The **data you have** is rarely the data you wish you had.

Model objective is nearly always a **proxy** for your business objectives

Some percentage of your customers may have a **bad experience**

The real world doesn’t stand still

Production ML and Change

Easy Problems

- Ground truth changes slowly (months, years)
- Model retraining driven by:
  - Model improvements, better data
  - Changes in software and/or systems
- Labeling
  - Curated datasets
  - Crowd-based

Harder Problems

- Ground truth changes faster (weeks)
- Model retraining driven by:
  - **Declining model performance**
  - Model improvements, better data
  - Changes in software and/or systems
- Labeling
  - Direct feedback
  - Crowd-based

Really Hard Problems

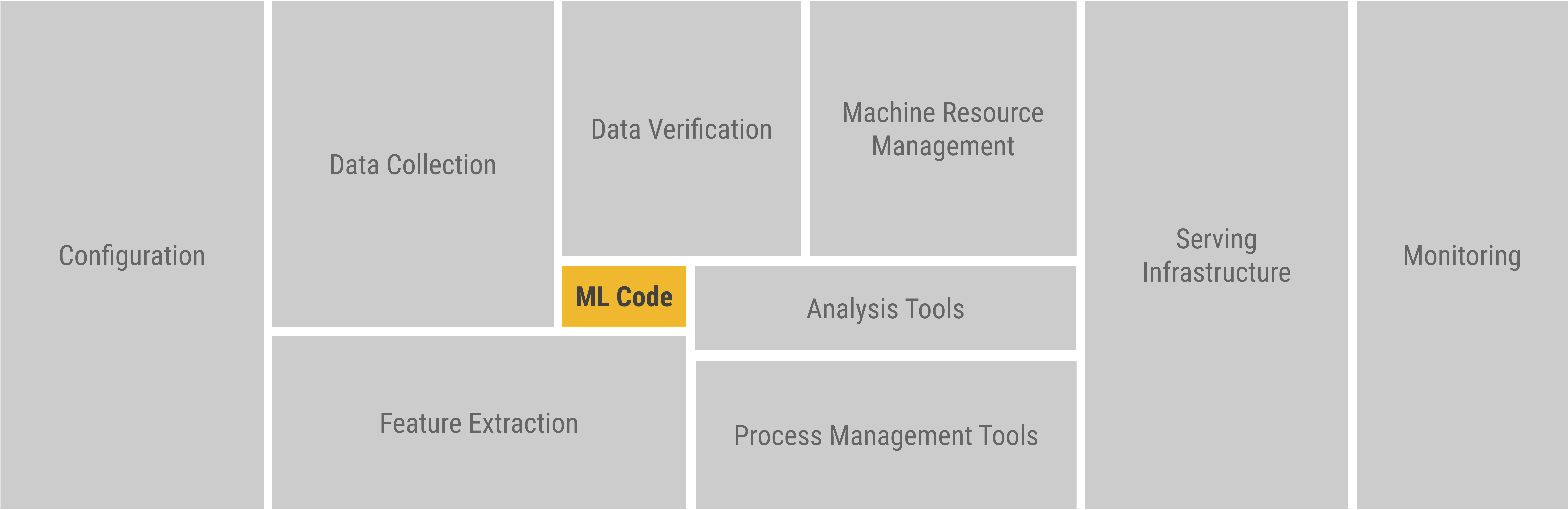
- Ground truth changes very fast (days, hours, min)
- Model retraining driven by:
  - Declining model performance
  - Model improvements, better data
  - Changes in software and/or systems
- Labeling
  - Direct feedback
  - Weak supervision

Machine Learning

In addition to training an amazing model ...

Modeling Code

… a production solution requires so much more



Tales From The Trenches

https://twitter.com/ginablaber/status/971450218095943681

TensorFlow Extended (TFX)

Tensorflow Extended (TFX)

Powers Alphabet's most important bets and products

LOON

nest

DeepMind

verily

"... we have re-tooled our machine learning platform to use TensorFlow. This yielded significant productivity gains while positioning ourselves to take advantage of the latest industry research."

Ranking Tweets with TensorFlow - Twitter

https://goo.gle/tf-twitter-rank

Production Machine Learning

Machine Learning Development

- Labeled data
- Feature space coverage
- Minimal dimensionality
- Maximum predictive data
- Fairness
- Rare conditions
- Data lifecycle management

Production Machine Learning

Machine Learning Development

- Labeled data
- Feature space coverage
- Minimal dimensionality
- Maximum predictive data
- Fairness
- Rare conditions
- Data lifecycle management

+

Modern Software Development

- Scalability
- Extensibility
- Configuration
- Consistency & Reproducibility
- Modularity
- Best Practices
- Testability
- Monitoring
- Safety & Security

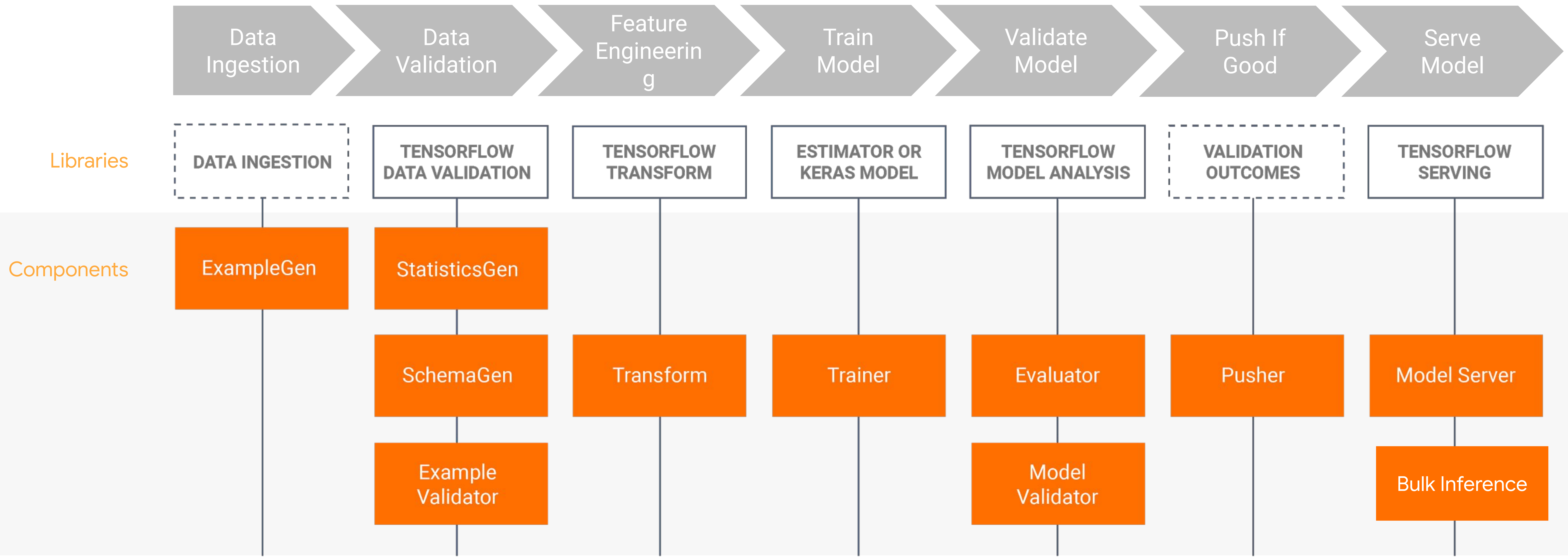
Production Machine Learning

"Hidden Technical Debt in Machine Learning Systems"

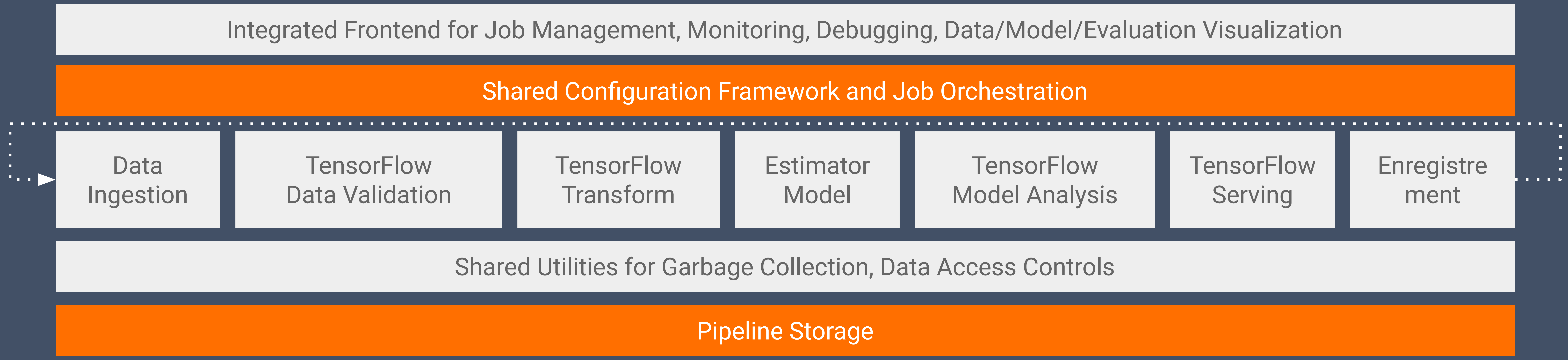
NIPS 2015

http://bit.ly/ml-techdebt

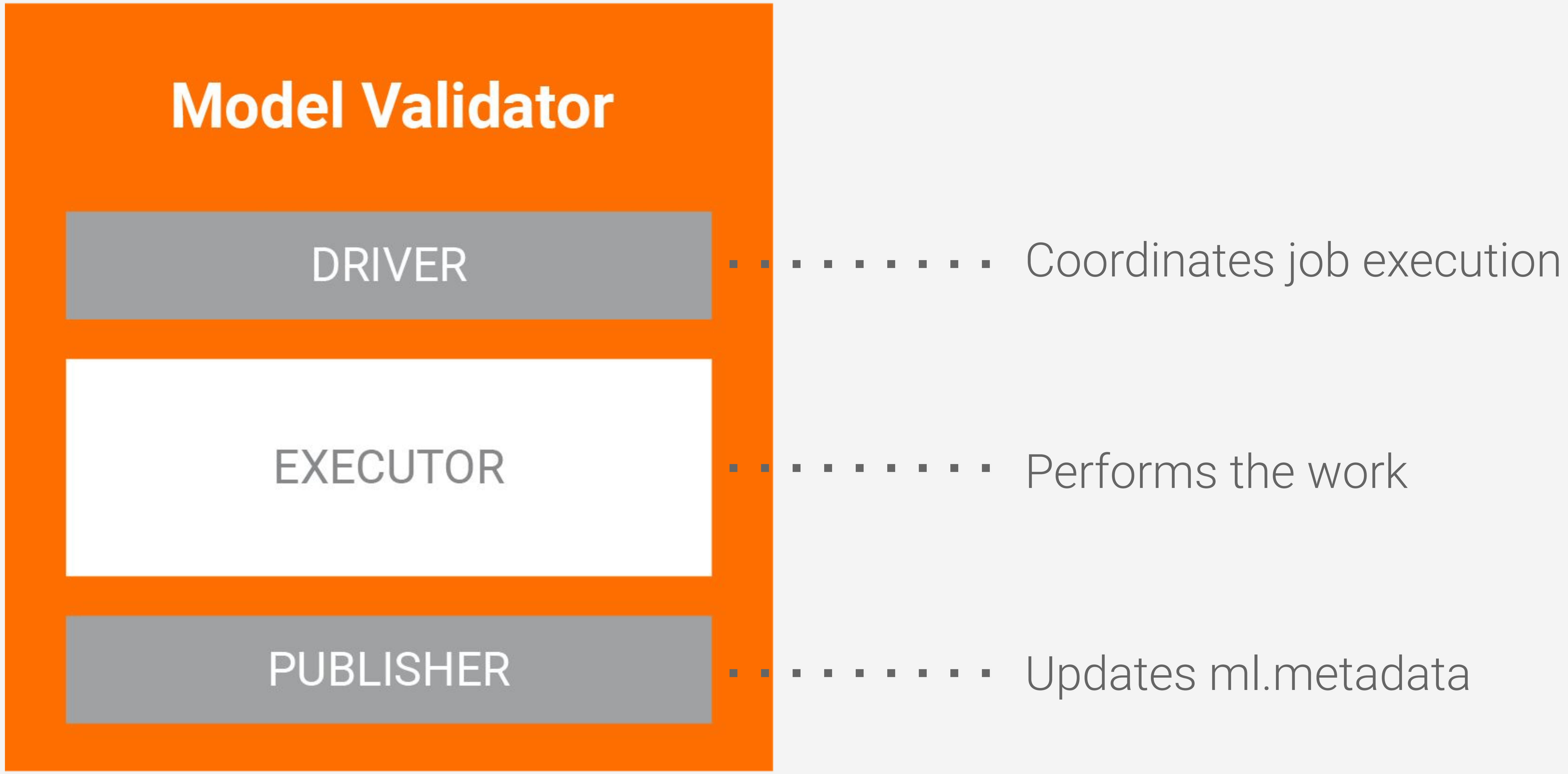
TFX Production Components



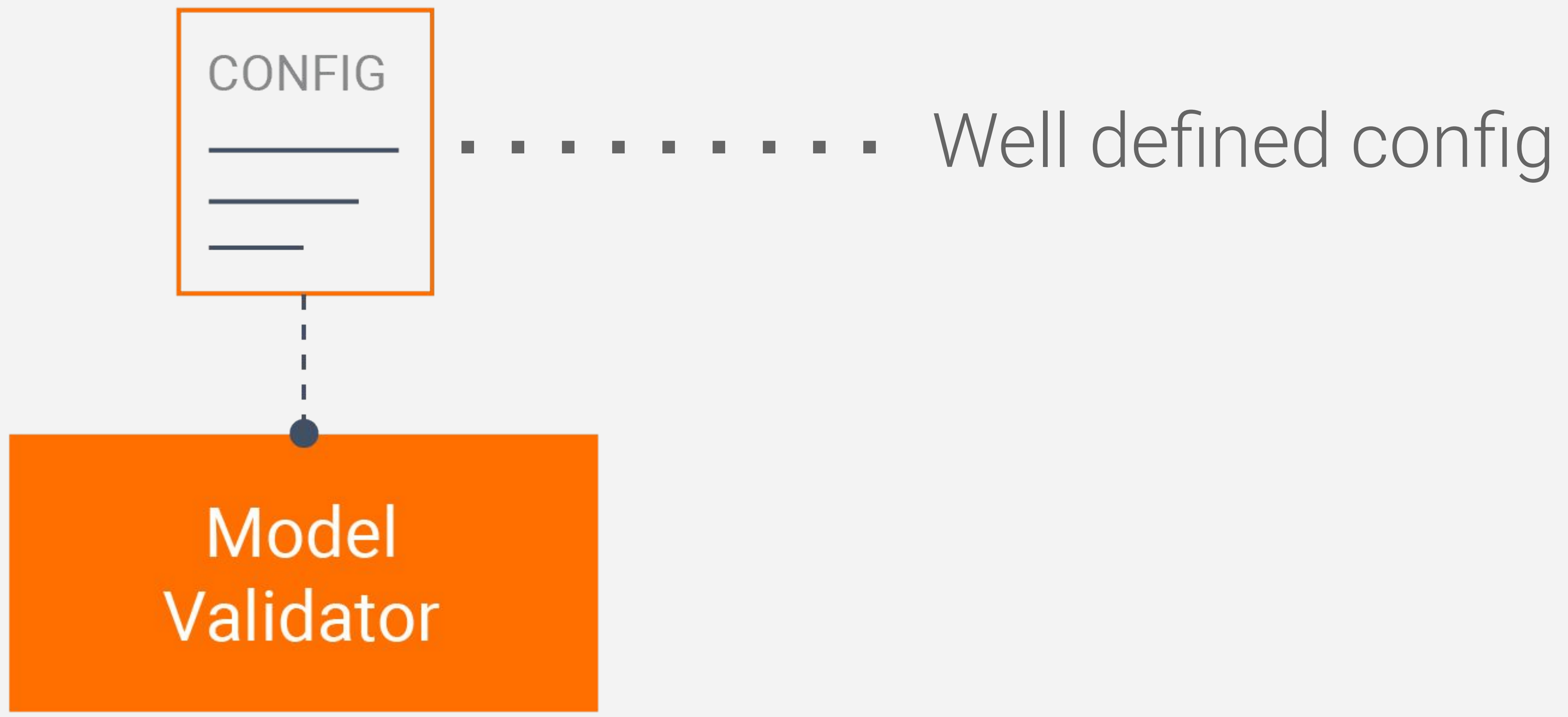
Horizontal Layers Coordinate Components



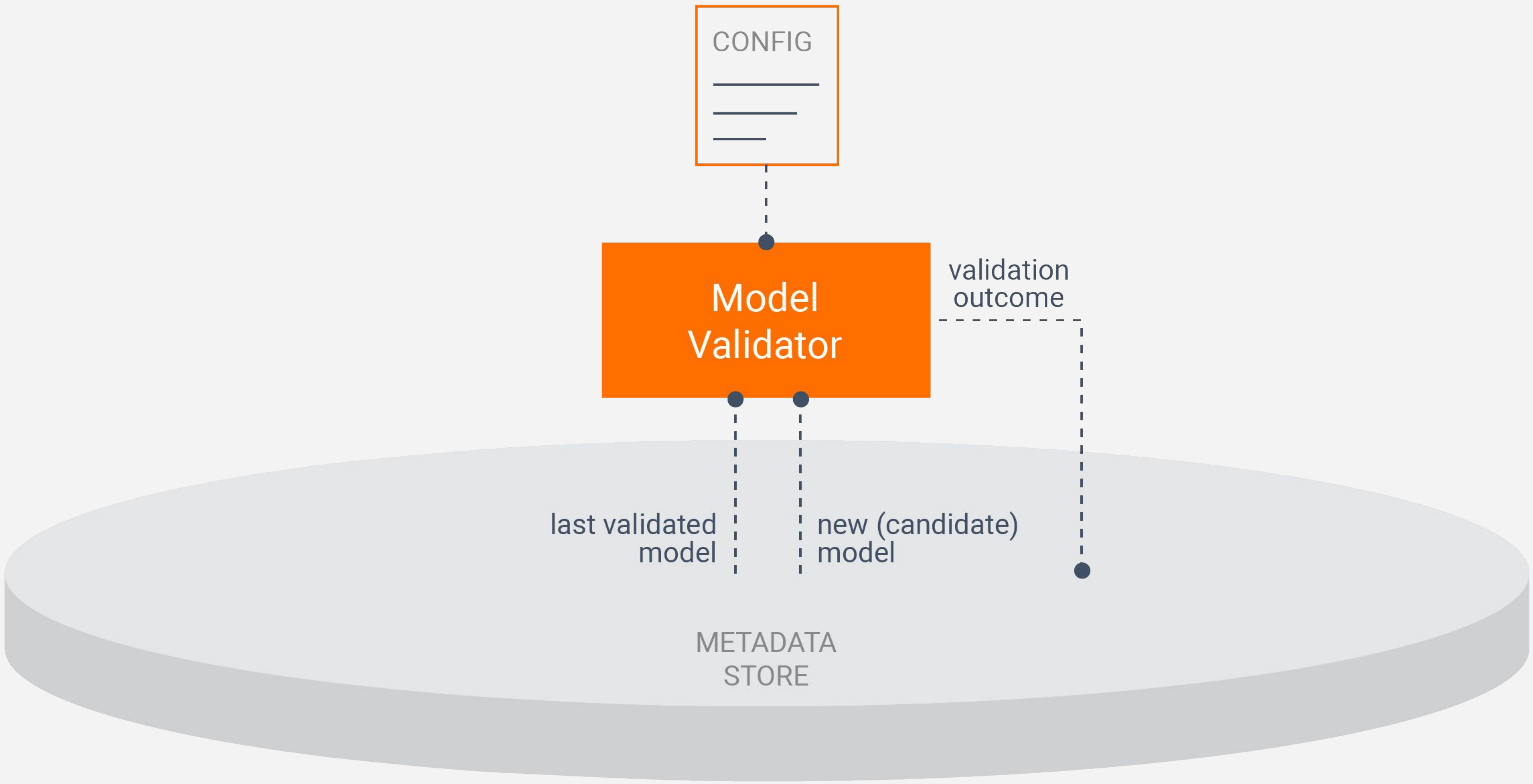
What is a TFX component?



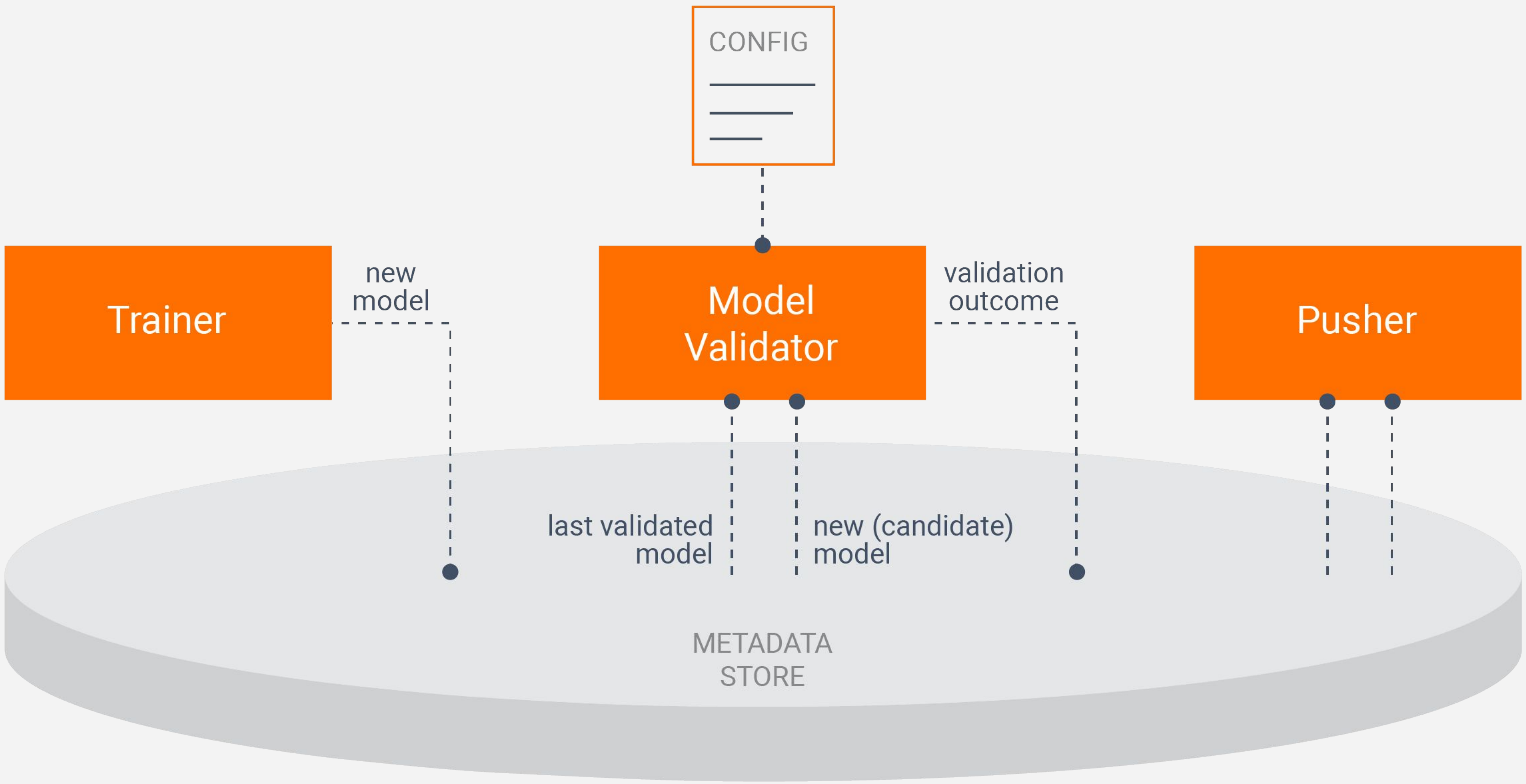
What makes a Component



What makes a Component?

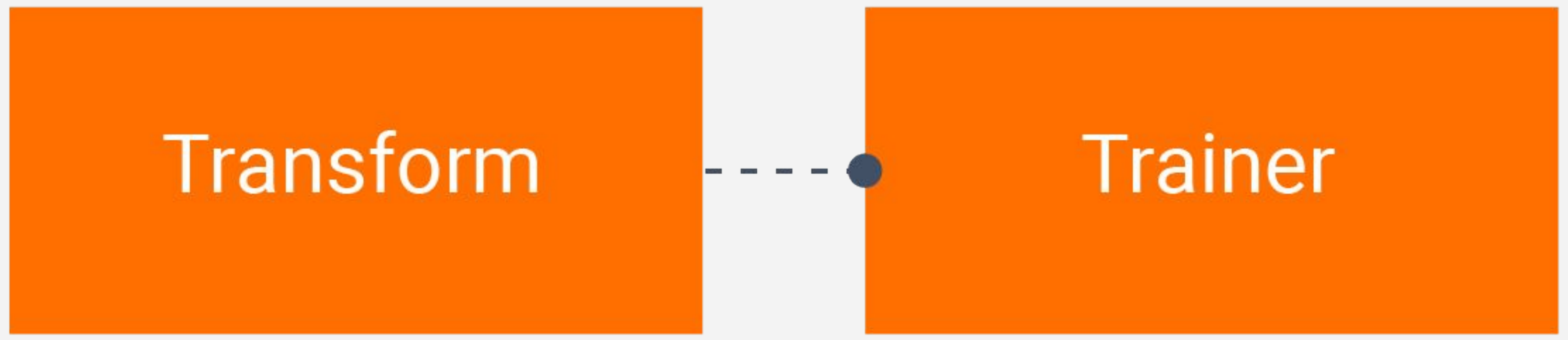


What makes a Component?

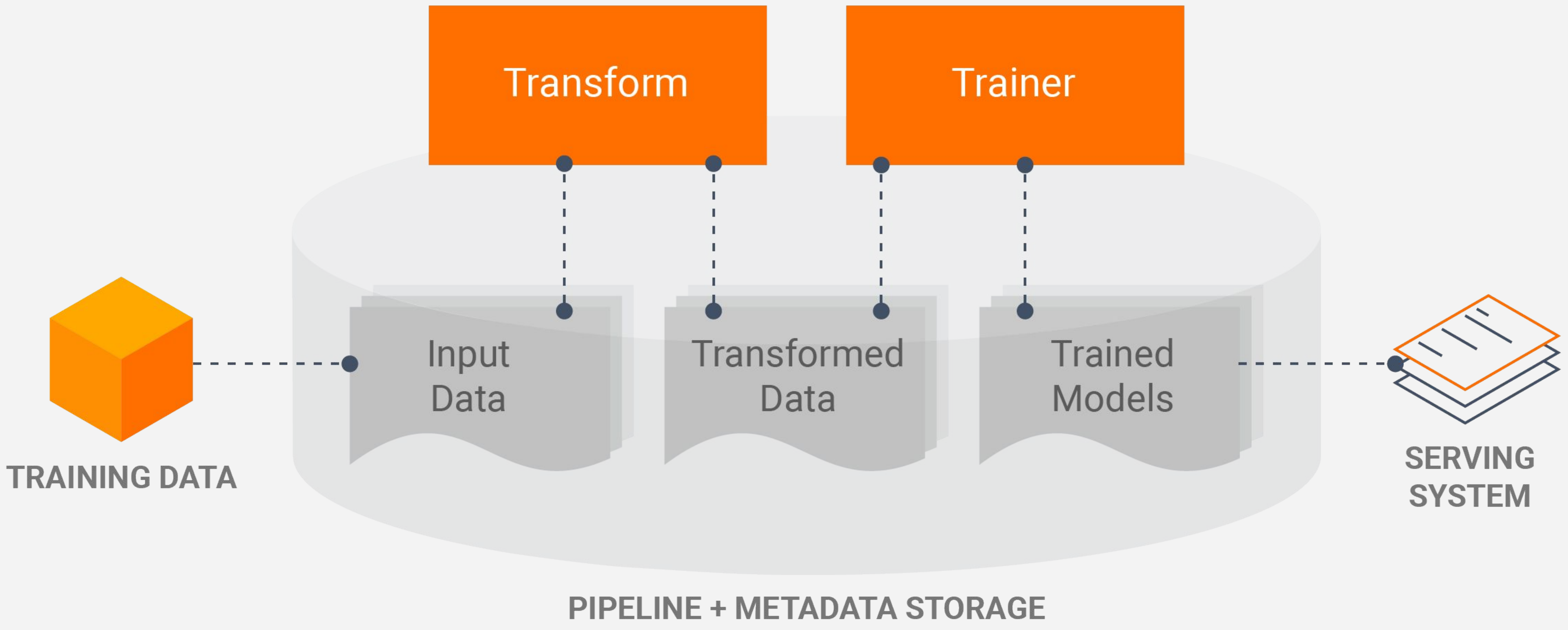


What makes a Component?

Orchestration Styles



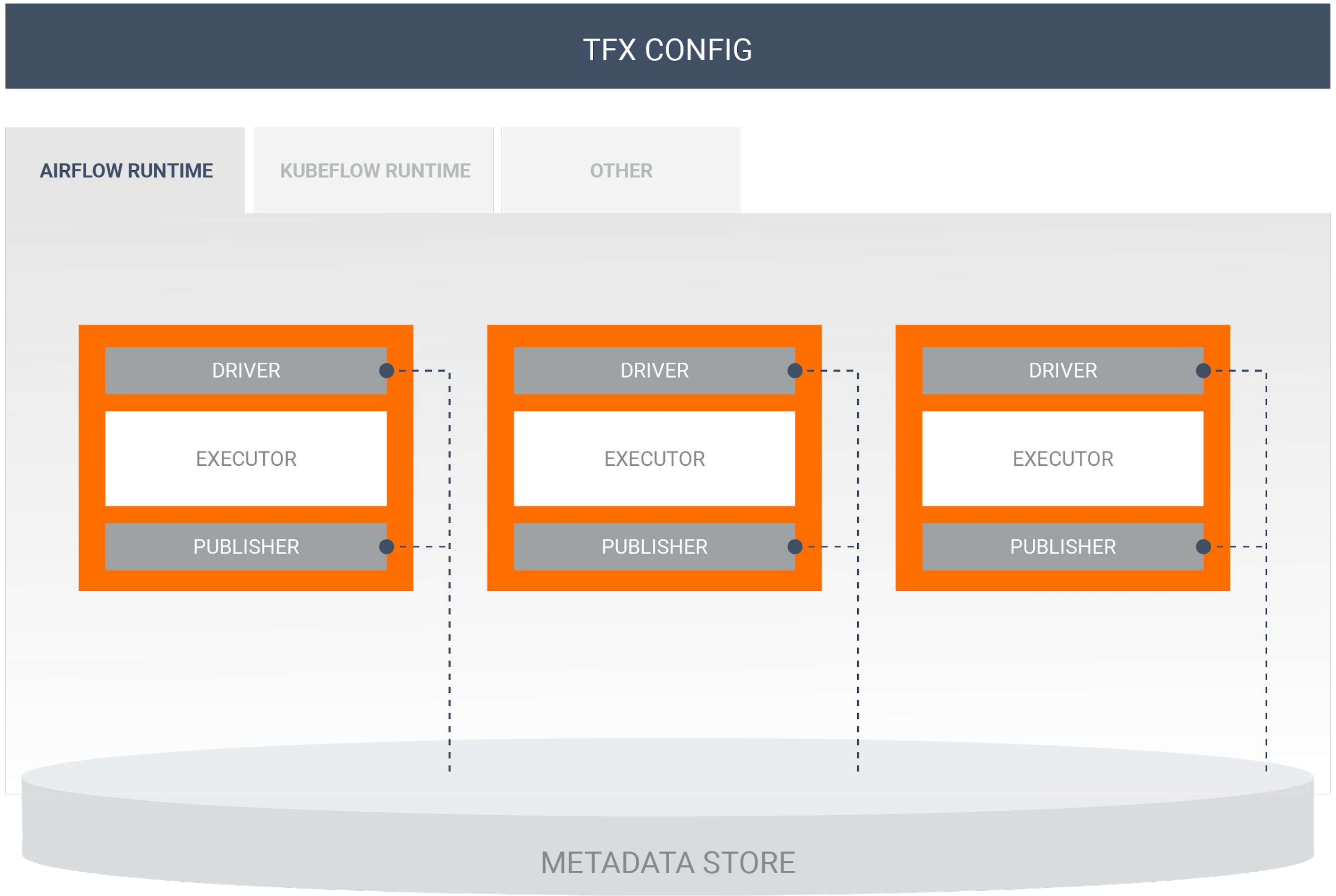
Task-Aware Pipelines



Task- and Data-Aware Pipelines

Metadata Store

TFX Orchestration

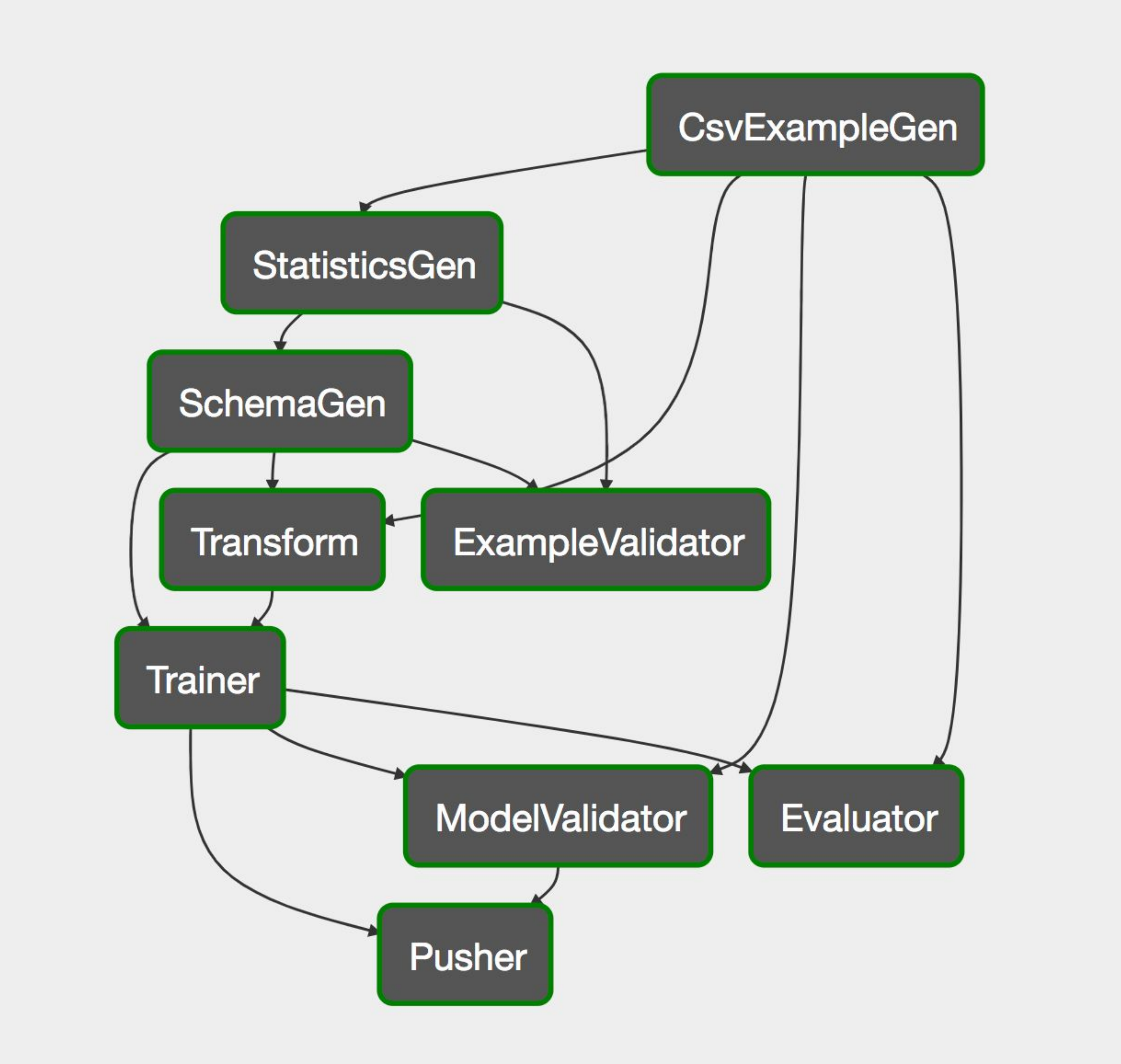


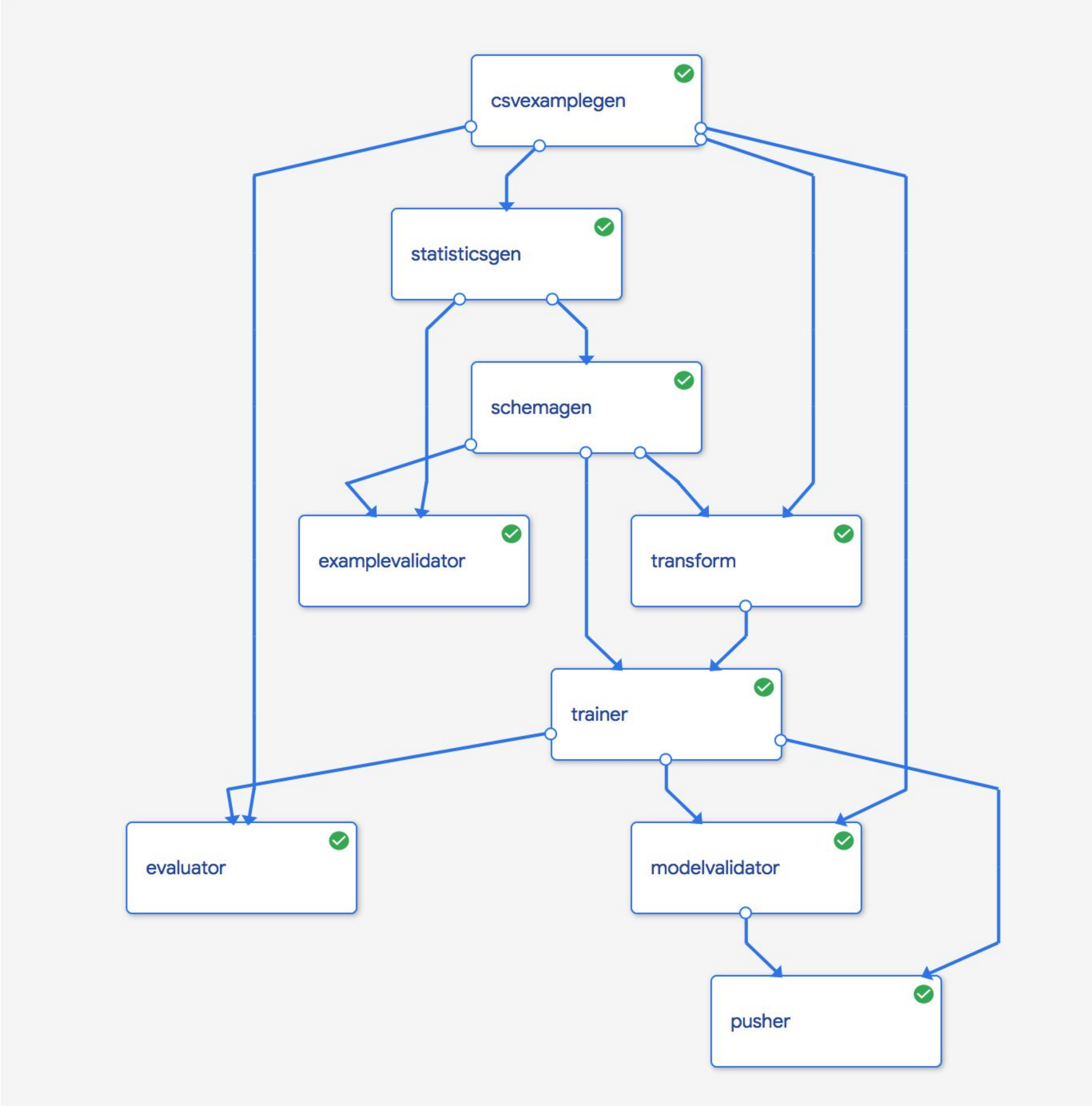
Bring your own Orchestrator

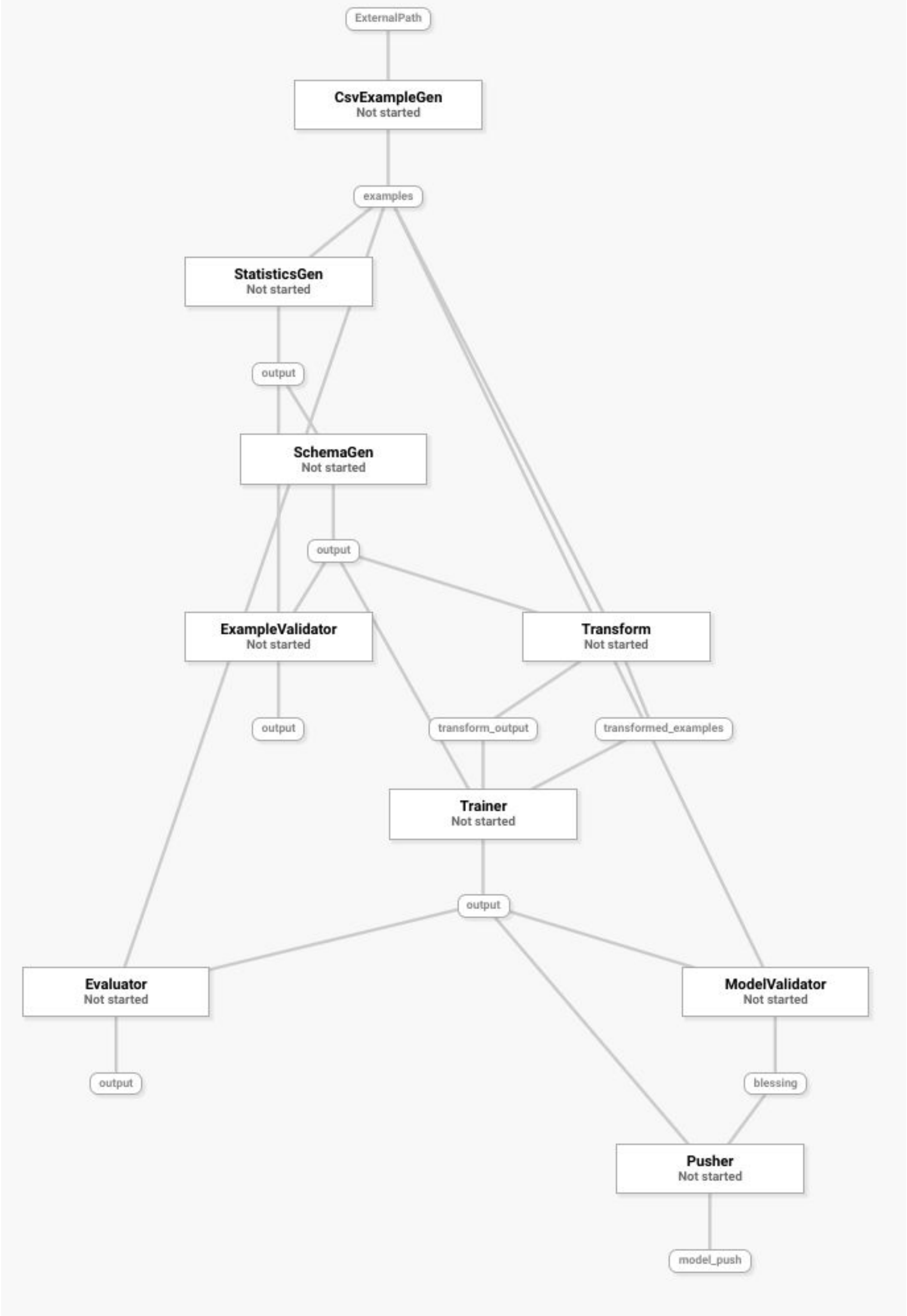
Flexible runtimes run components in the proper order using orchestration systems such as Airflow, Kubeflow, or Beam

Orchestrators and DAGs

Apache Airflow	Kubeflow Pipelines	Apache Beam







TFX and Kubeflow Pipelines

TensorFlow Extended (TFX)

- Open-source version of what Google uses internally for Production ML
- Currently supported orchestrators:
  - Kubeflow
  - Apache Airflow
  - Apache Beam
  - We're adding more
  - You can add more

Kubeflow Pipelines

Kubeflow

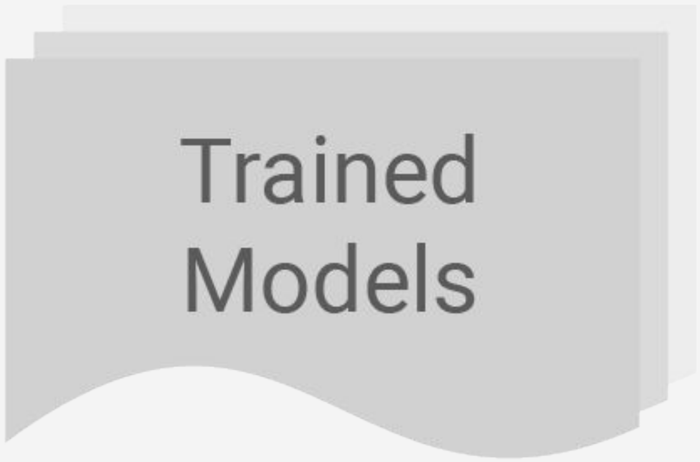
- Metadata tracking + caching enabled, ability to resume pipelines from crashes.
- Containers as custom components
- Orchestrate existing R/C++/Scala components and get metadata tracking + caching
- Artifact provenance and lineage visualized
- Deploy using Marketplace
- CloudSQL can be used to persist pipeline metadata across clusters

TFX Orchestration in a Notebook

- Experimental environment for iterative development
- Build up your pipeline iteratively in a Jupyter / Colab notebook and export to production with minimal changes
- InteractiveContext object handles component execution and artifact visualization
- In production, you would use Airflow, Kubeflow, or similar

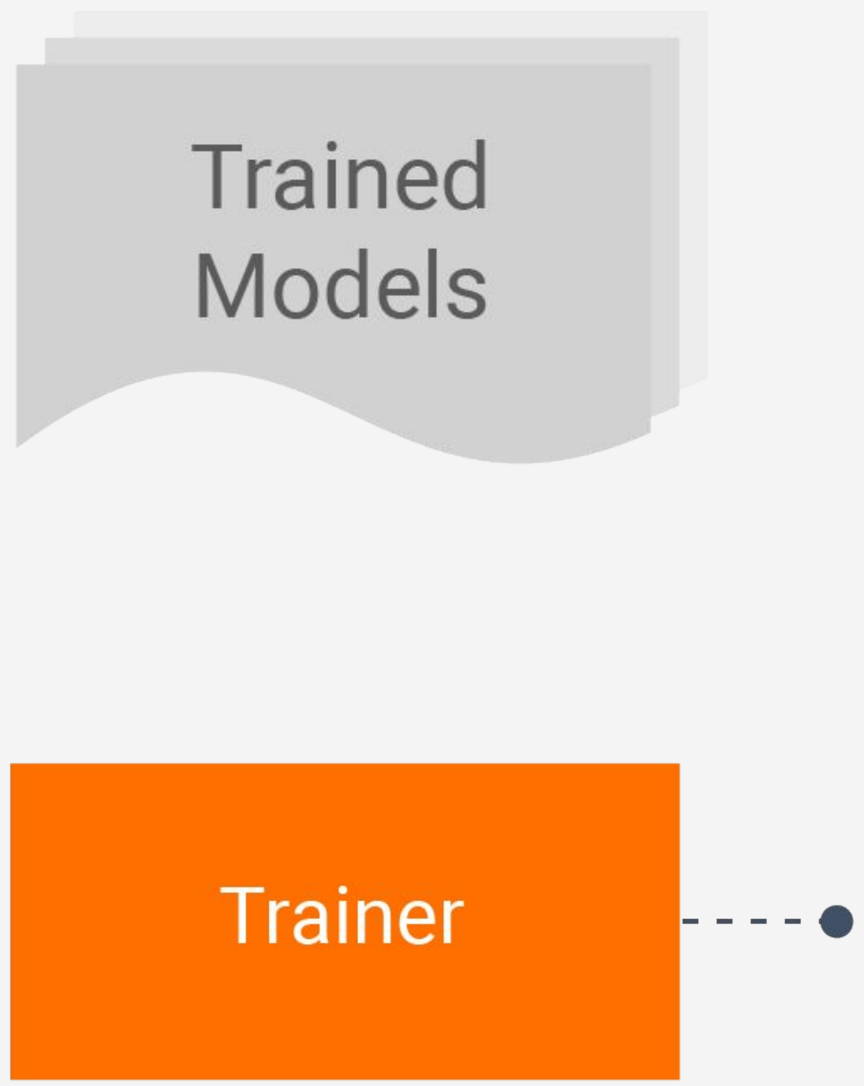
```
context = InteractiveContext()

component = MyComponent(...)
context.run(component)
context.show(component.outputs['my_output'])
```

Metadata Store



What is in Metadata Store?

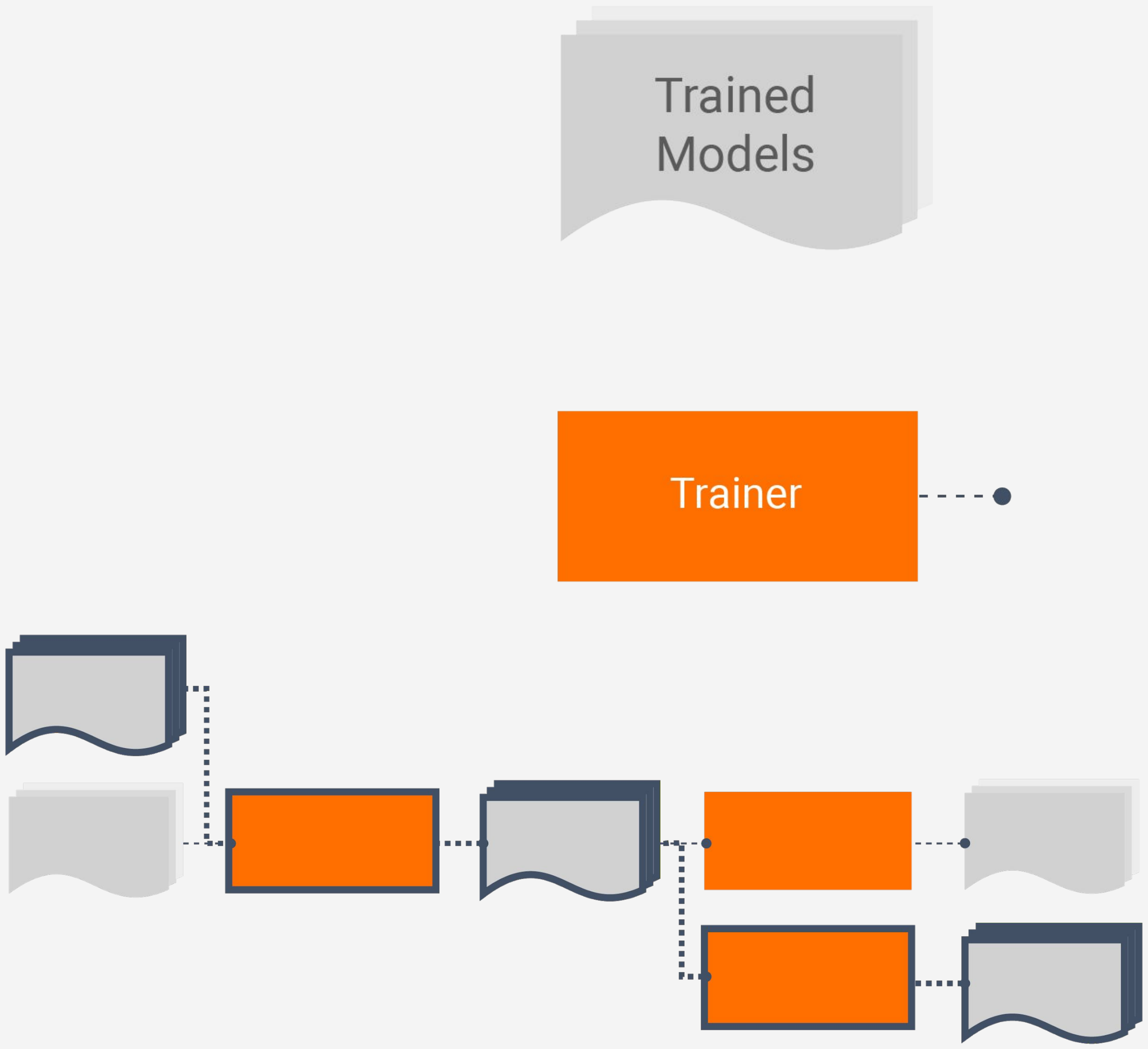
Type definitions of Artifacts and their Properties



What is in Metadata Store?

Type definitions of Artifacts and their Properties

Execution Records (Runs) of Components



What is in Metadata Store?

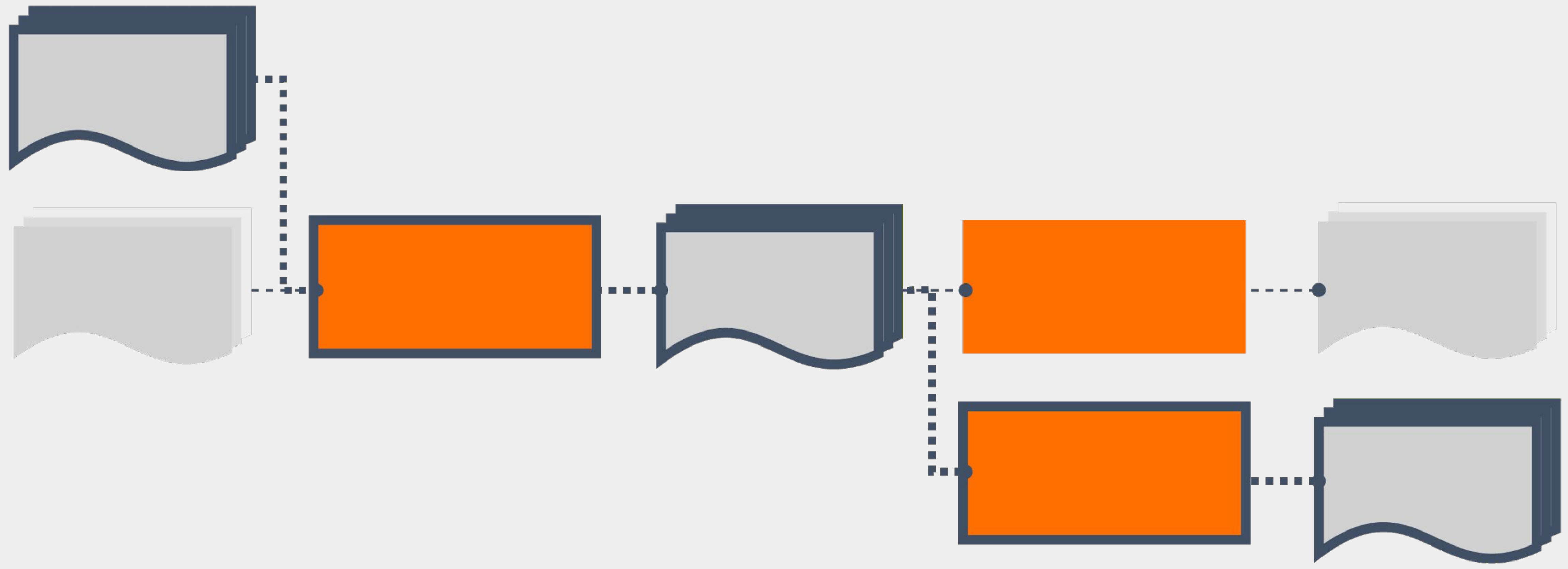
Type definitions of Artifacts and their Properties

Execution Records (Runs) of Components

Data Provenance Across All Executions

Metadata-Powered Functionality

Find out which data a model was trained on

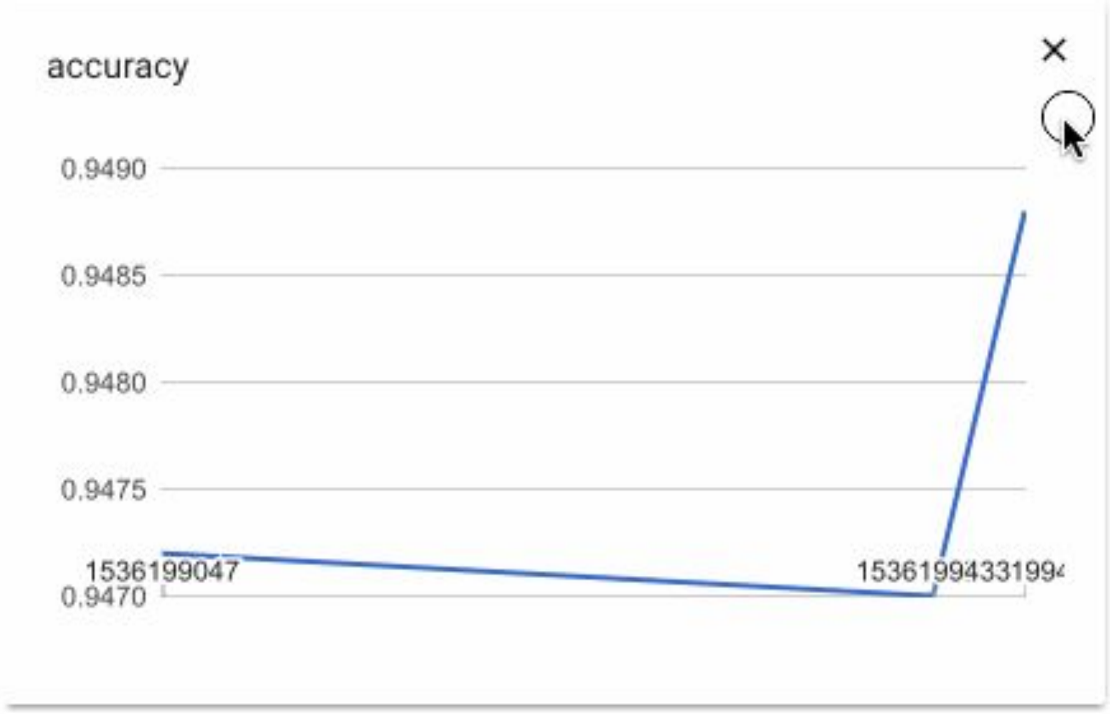


Metadata-Powered Functionality

Compare previous model runs

```
eval_results = tfma.make_eval_results([tfma_result_1, tfma_result_2, tfma_result_3],
                                      tfma.constants.MODEL_CENTRIC_MODE)
tfma.view.render_time_series(eval_results, OVERALL_SLICE_SPEC)
```

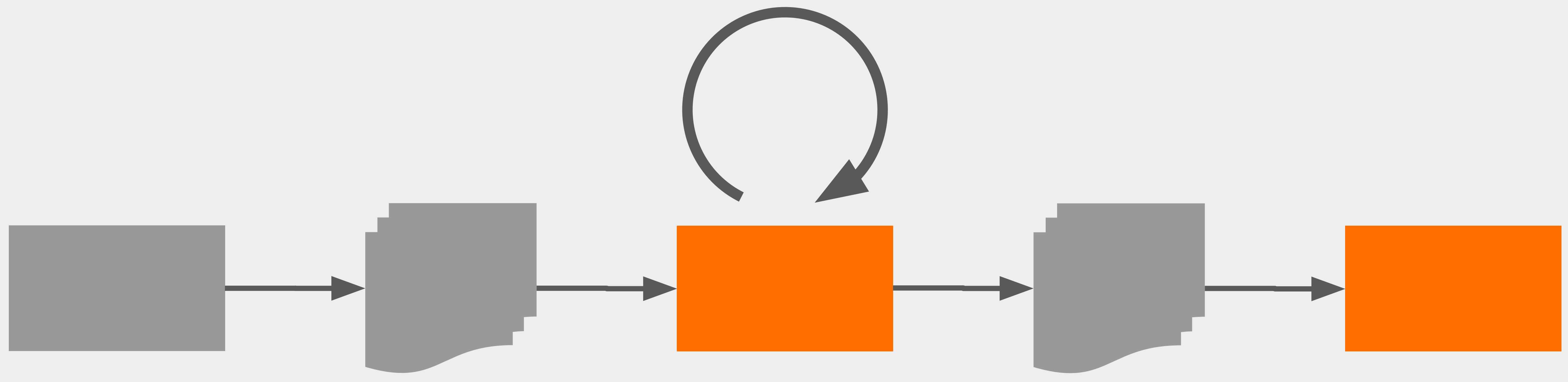
Add metric series



Model	Data	accuracy	accuracy_baseline	auc	auc_precision_recall	average_loss	label/mean
1536199479	data.csv	0.94880	0.94220	0.93168	0.98516	0.13980	0.94220
1536199433	data.csv	0.94700	0.94220	0.93165	0.98170	0.13979	0.94220
1536199047	data.csv	0.94720	0.94220	0.92914	0.99480	0.14103	0.94220

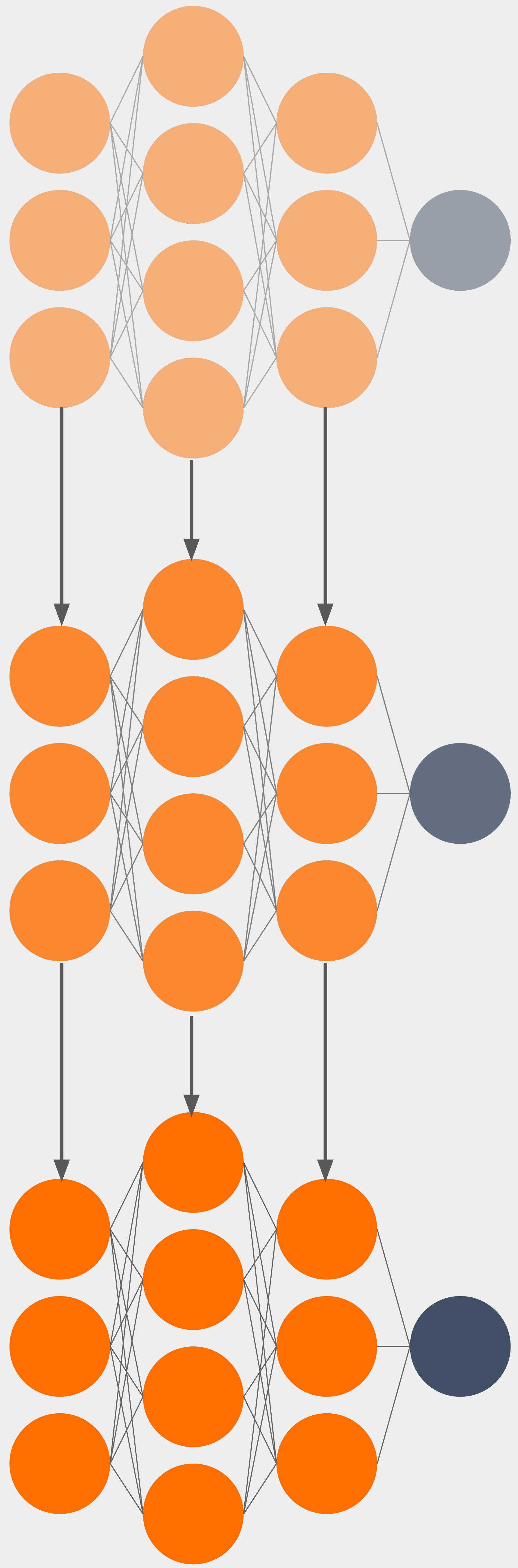
Metadata-Powered Functionality

Re-use previously computed outputs



Metadata-Powered Functionality

Carry-over state from previous model runs



Distributed Pipeline Processing: Apache Beam

What is Apache Beam?

- A unified **batch** and stream distributed processing API
- A set of **SDK frontends**: Java, **Python**, Go, Scala, SQL
- A set of **Runners** which can execute Beam jobs into various backends: **Local, Apache Flink**, **Apache Spark**, Apache Gearpump, **Apache Samza**, Apache Hadoop, **Google Cloud Dataflow**, ...

Apache Beam

Java

```
input.apply(
  Sum.integersPerKey())
```

Python

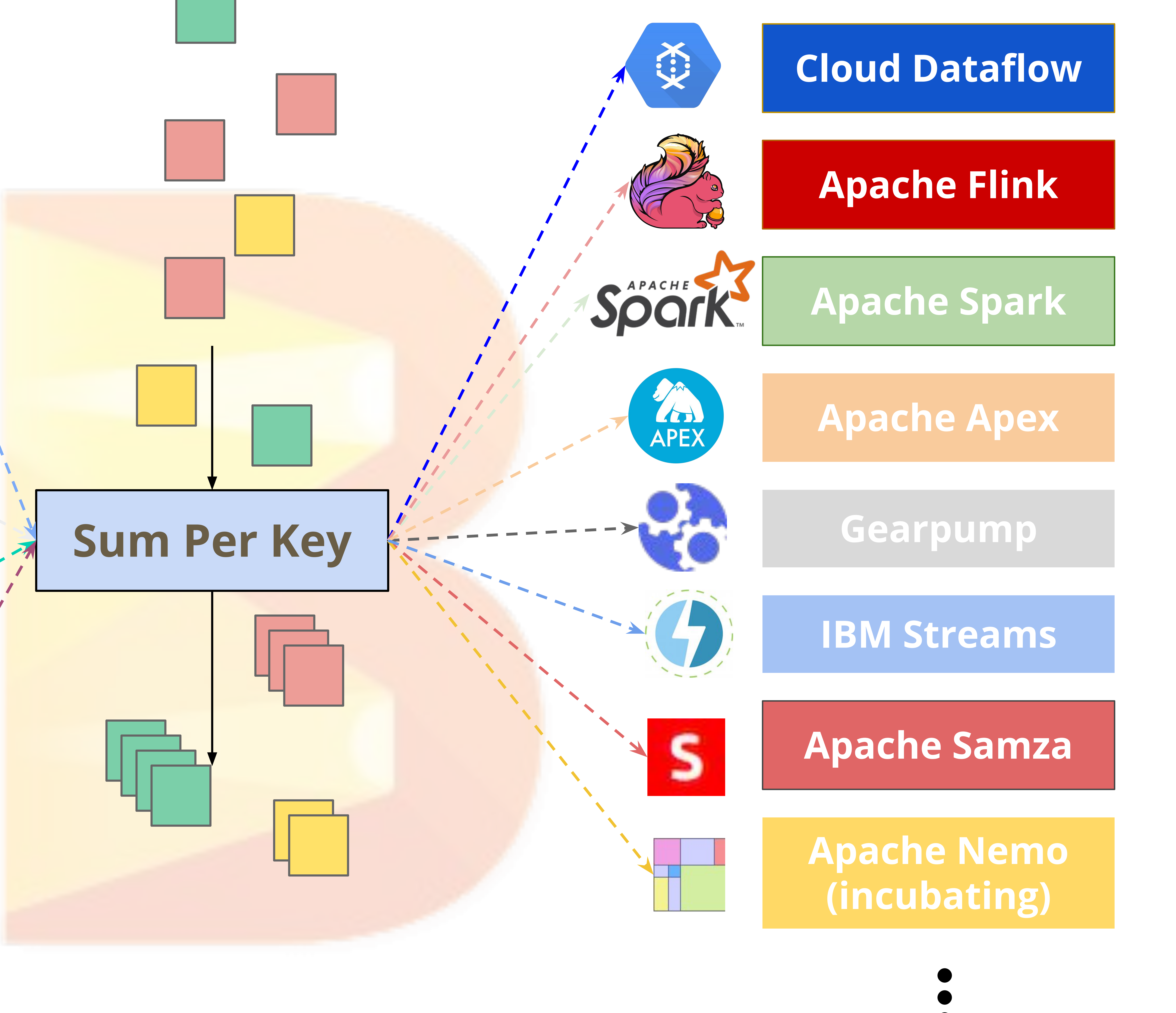
```
input | Sum.PerKey()
```

Go

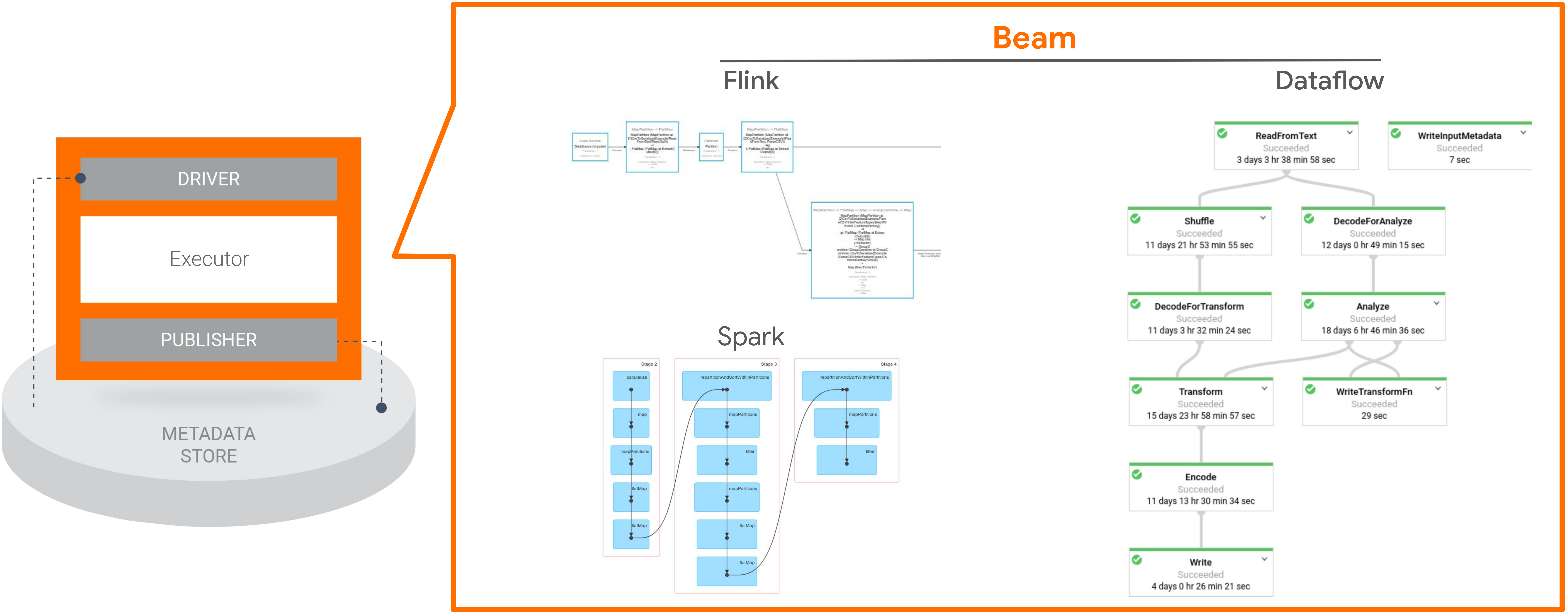
```
stats.Sum(s, input)
```

SQL

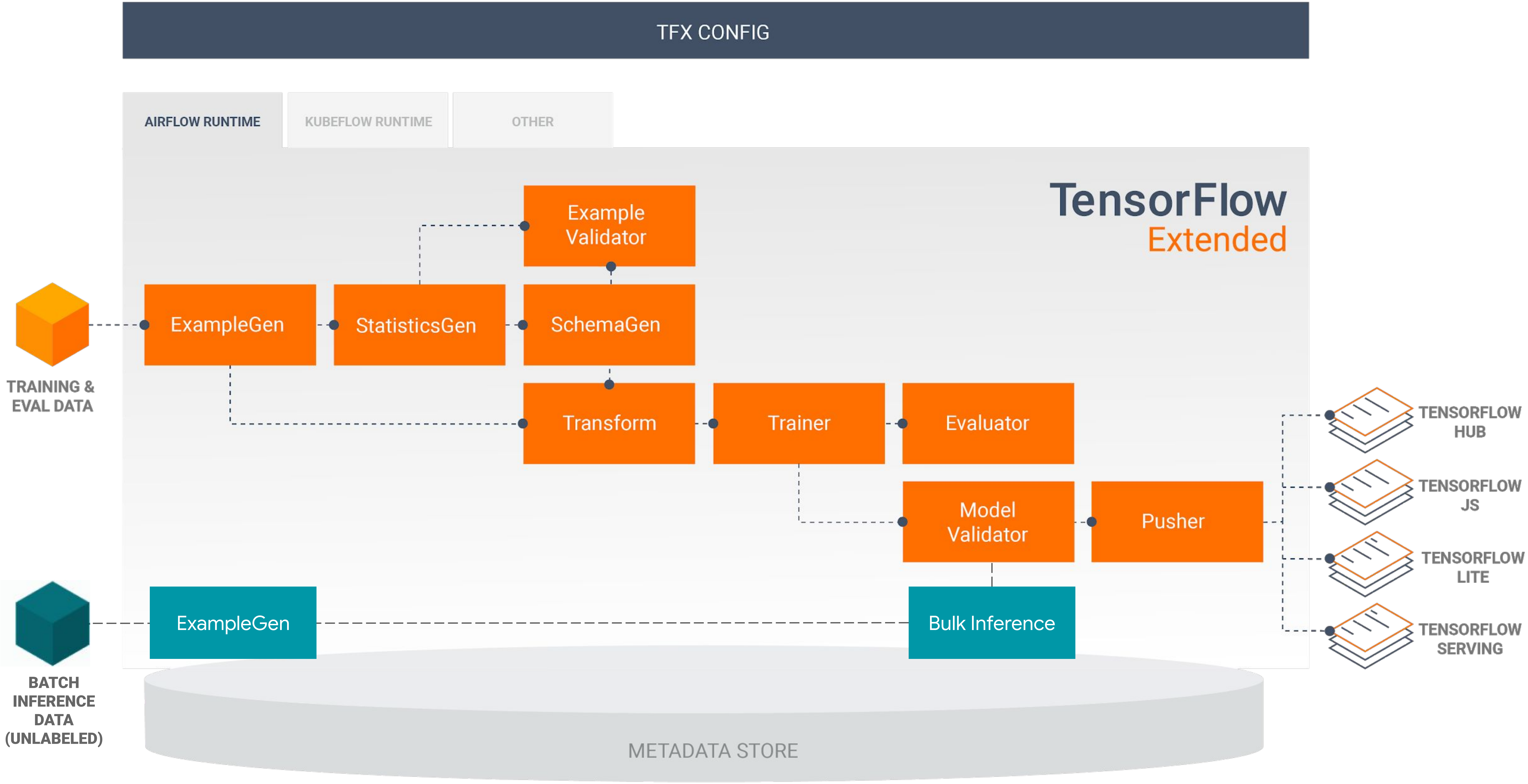
```
SELECT key, SUM(value)
FROM input GROUP BY key
```



TFX Components & Beam

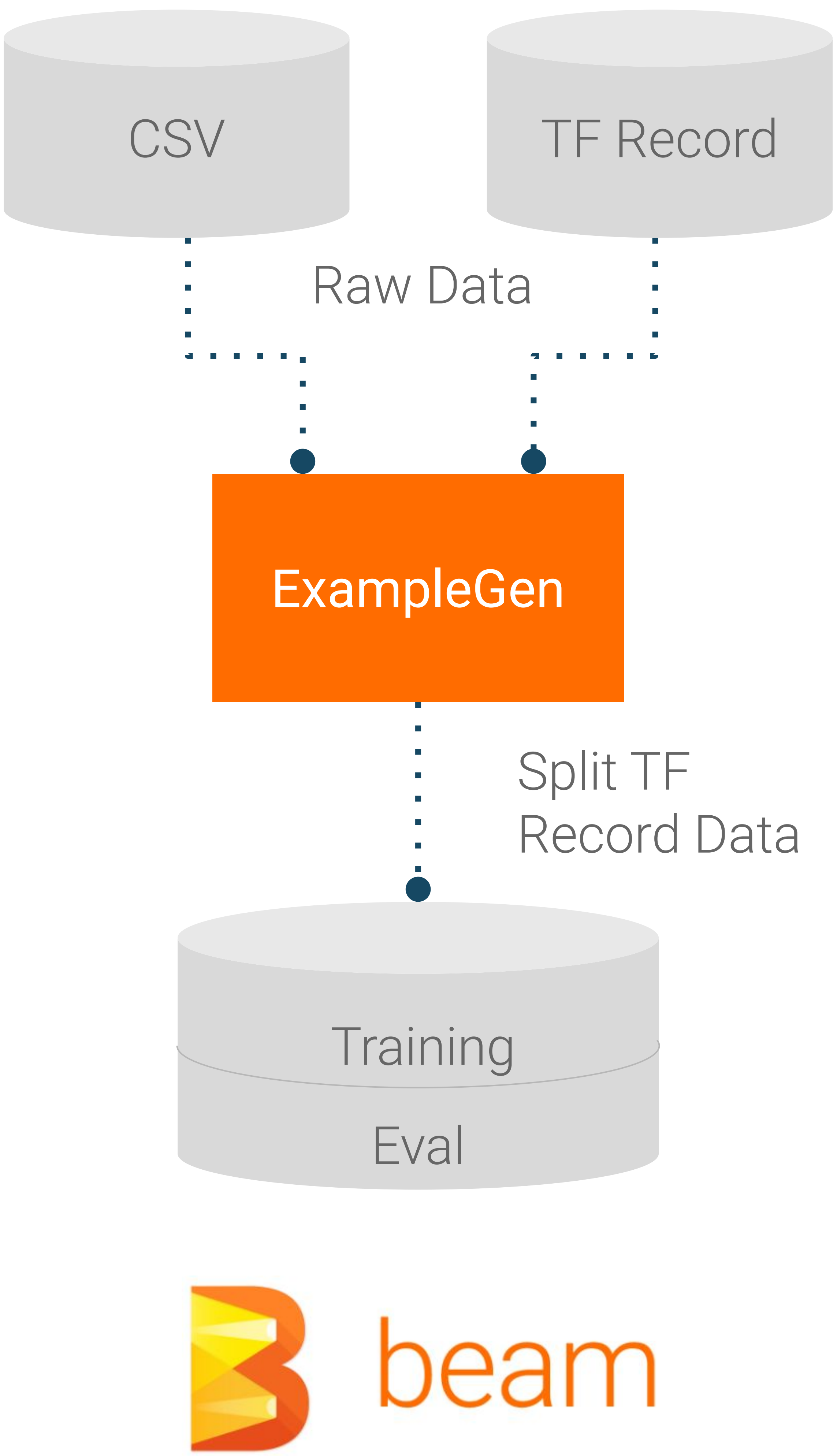


TFX Standard Components


TFX CONFIG
AIRFLOW RUNTIME
KUBEFLOW RUNTIME
OTHER
TensorFlow
Extended
Example Validator
ExampleGen
StatisticsGen
SchemaGen
TRAINING & EVAL DATA
Transform
Trainer
Evaluator
Model Validator
Pusher
TENSORFLOW HUB
TENSORFLOW JS
TENSORFLOW LITE
TENSORFLOW SERVING
ExampleGen
Bulk Inference
BATCH INFERENCE DATA (UNLABELED)
METADATA STORE

Component: ExampleGen

Inputs and Outputs

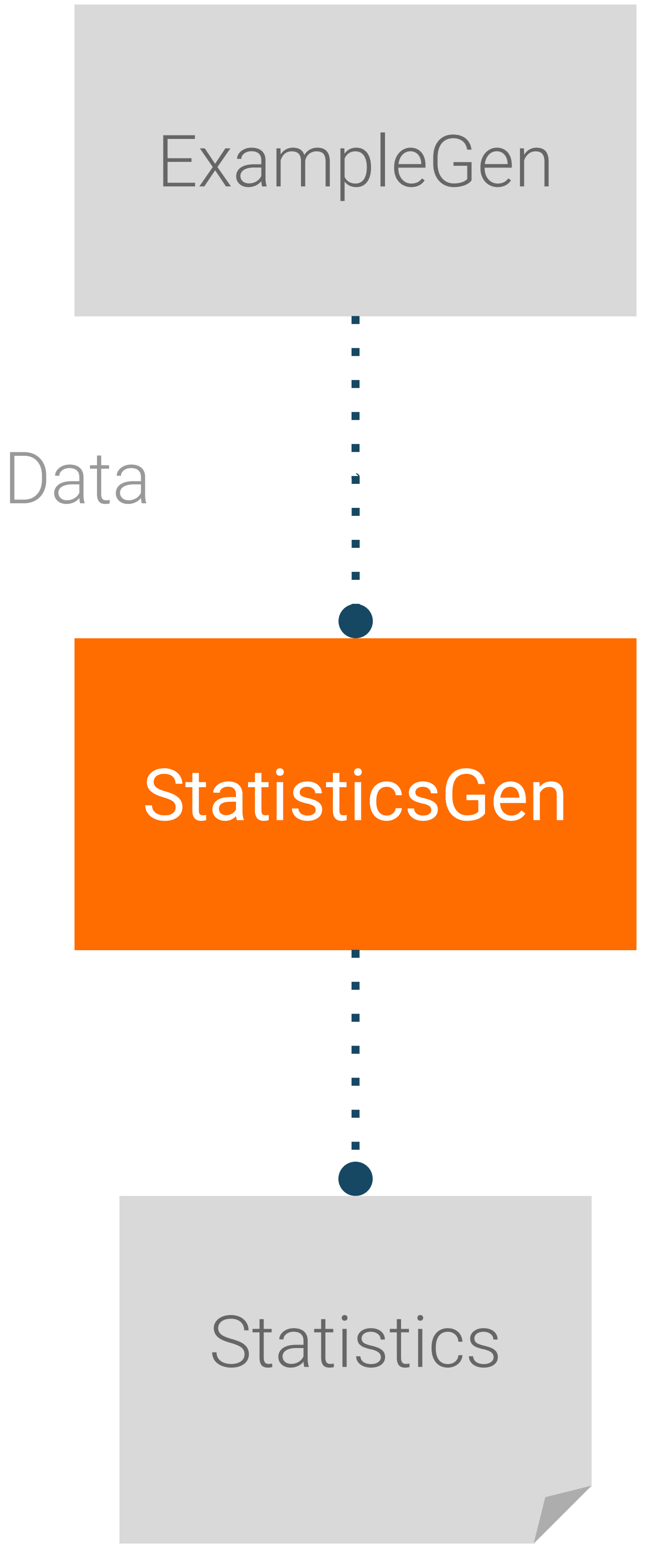


Configuration

```
example_gen = CsvExampleGen(input_base=external_input(data_root))
```

Component: StatisticsGen

Inputs and Outputs

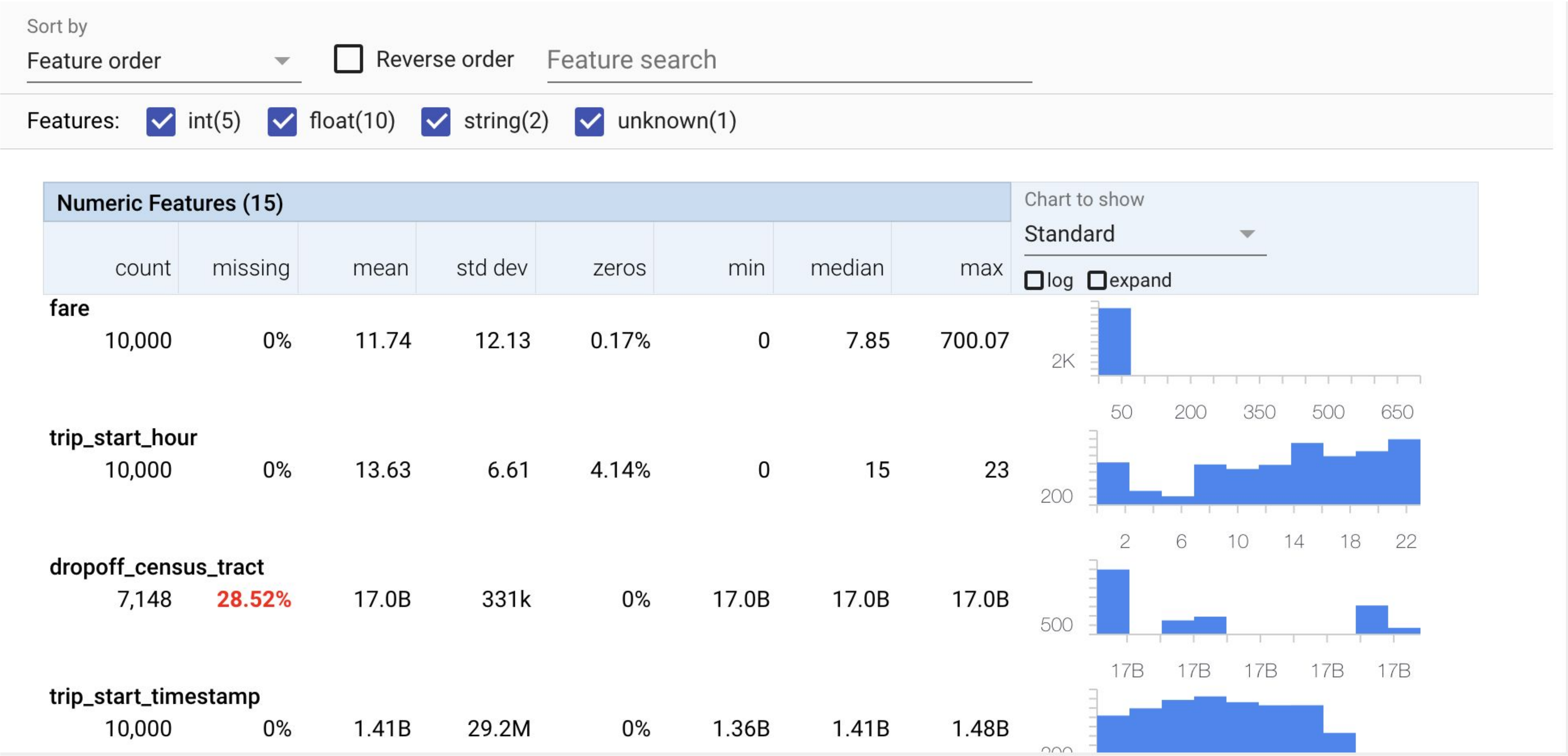


beam

Configuration

```
statistics_gen = StatisticsGen(
    input_data=example_gen.outputs['examples'])
```

Visualization



	count	missing	mean	std dev	zeros	min	median	max
fare	10,000	0%	11.74	12.13	0.17%	0	7.85	700.07
trip_start_hour	10,000	0%	13.63	6.61	4.14%	0	15	23
dropoff_census_tract	7,148	28.52%	17.0B	331k	0%	17.0B	17.0B	17.0B
trip_start_timestamp	10,000	0%	1.41B	29.2M	0%	1.36B	1.41B	1.48B

Analyzing Data with TensorFlow Data Validation

Sort by

Feature order

Reverse order

Feature search (regex enabled)

Features: int(5) float(5) variable-length floats(5) string(1) fixed-length strings(1) variable-length strings(1)

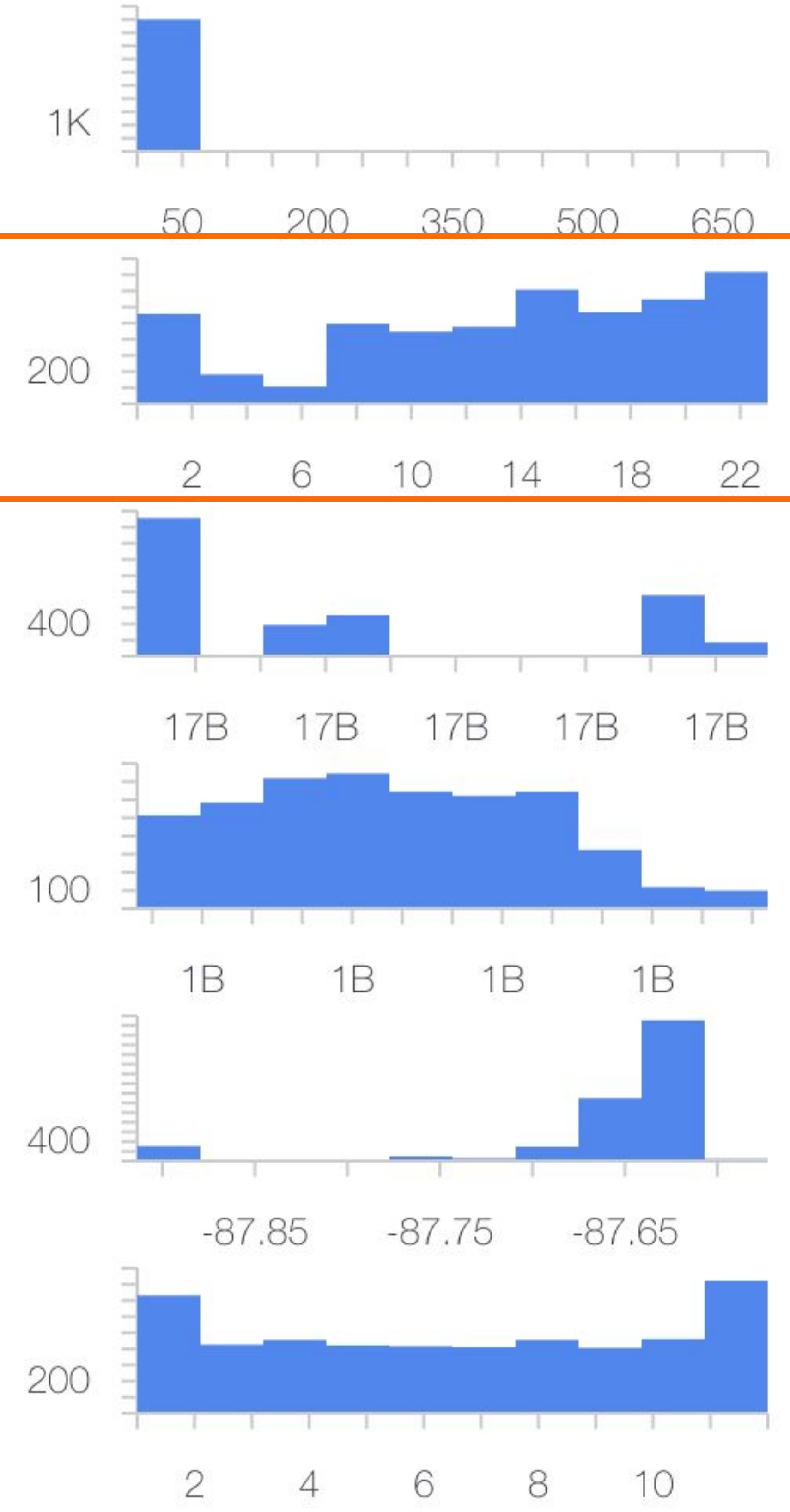
Numeric Features (15)

	count	missing	mean	std dev	zeros	min	median	max
fare	5,009	0%	11.97	14.12	0.12%	0	7.85	700.07
trip_start_hour	5,009	0%	13.51	6.73	4.13%	0	15	23
dropoff_census_tract	5,009	0%	17.0B	328k	0%	17.0B	17.0B	17.0B
trip_start_timestamp	5,009	0%	1.41B	29.0M	0%	1.36B	1.41B	1.48B
pickup_longitude	5,009	0%	-87.66	0.07	0%	-87.91	-87.63	-87.57
trip_start_month	5,009	0%	6.6	3.4	0%	1	7	12

Chart to show

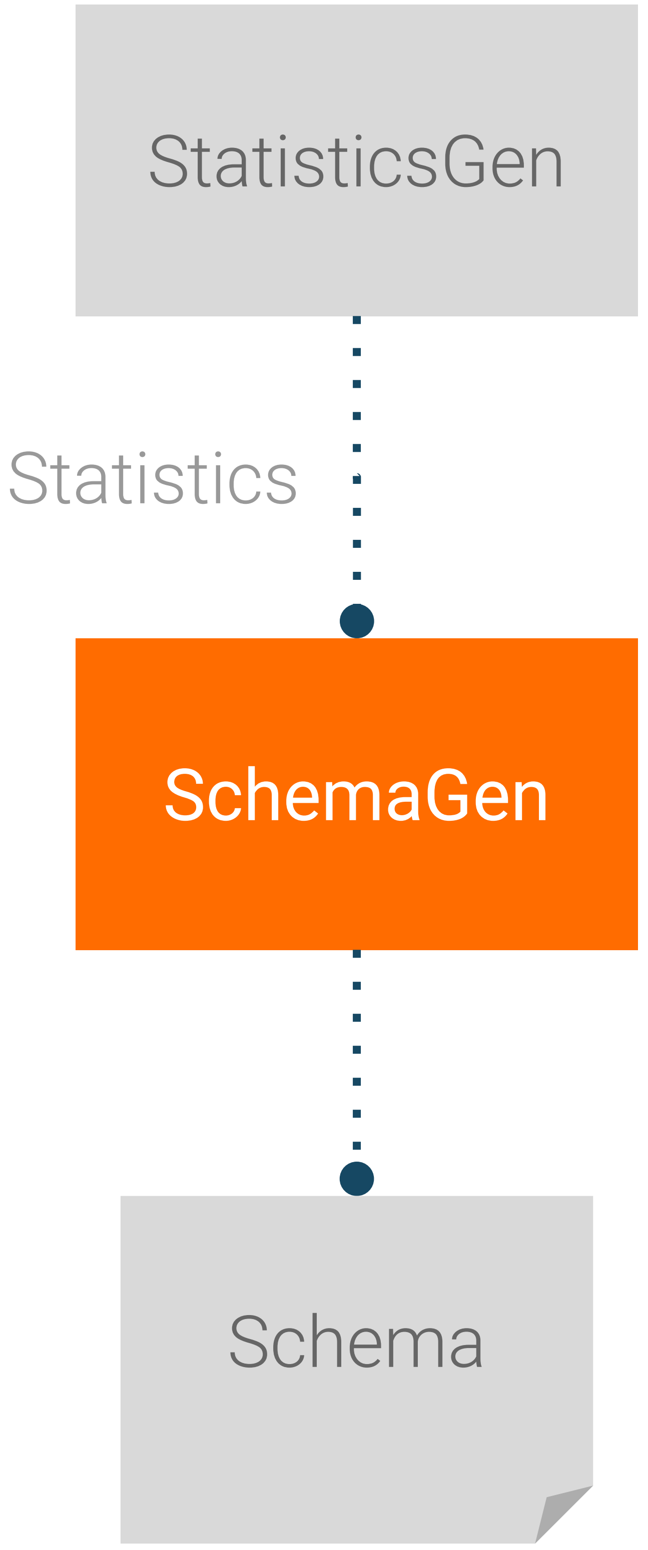
Standard

log expand



Component: SchemaGen

Inputs and Outputs



Configuration

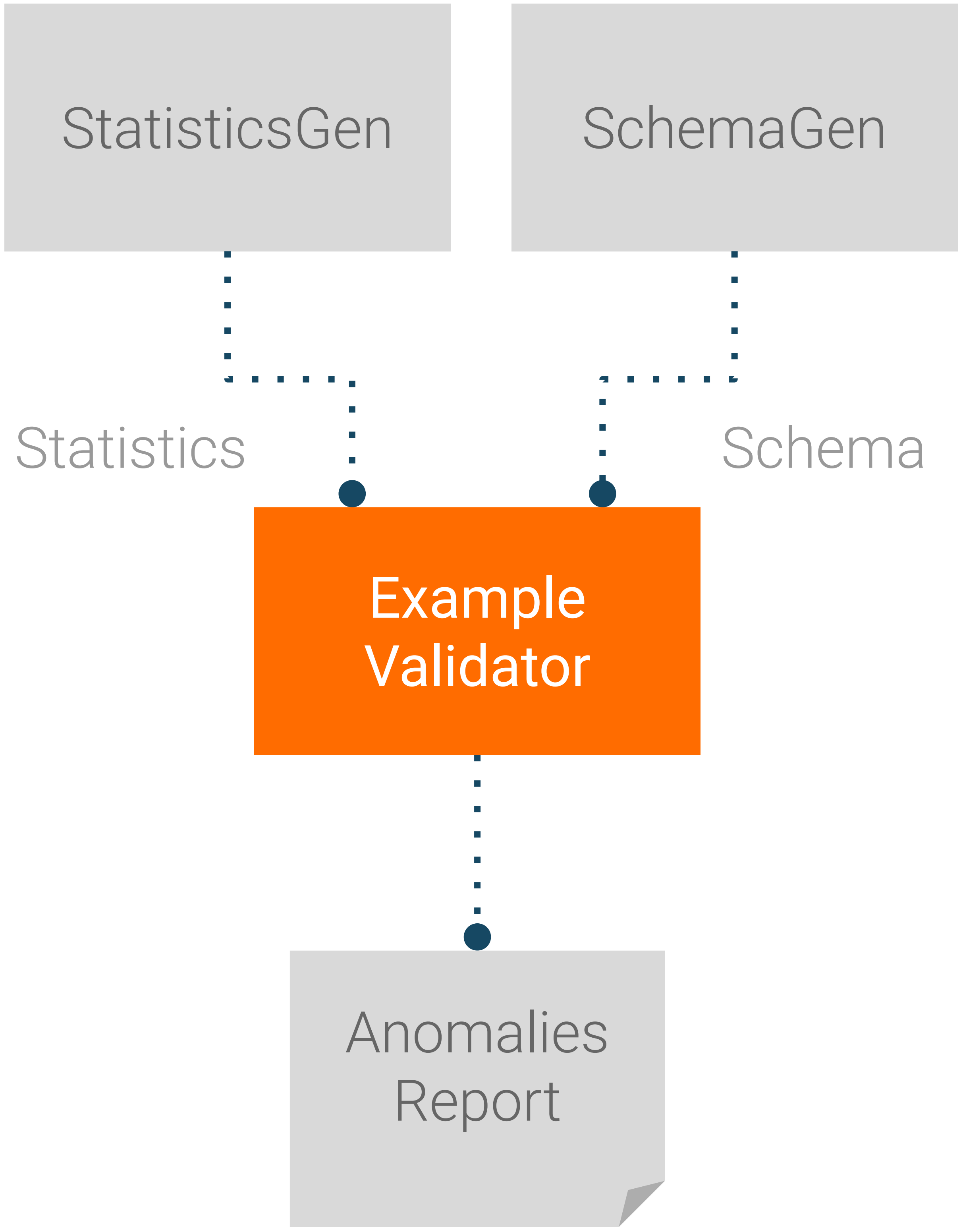
```
infer_schema =
    SchemaGen(statistics=statistics_gen.outputs['statistics'],
              infer_feature_shape=False)
```

Visualization

Feature name	Type	Presence	Valency	Domain
'fare'	FLOAT	required	single	-
'trip_start_hour'	INT	required	single	-
'pickup_census_tract'	BYTES	optional		-
'dropoff_census_tract'	FLOAT	optional	single	-
'company'	STRING	optional	single	'company'

Component: ExampleValidator

Inputs and Outputs



Configuration

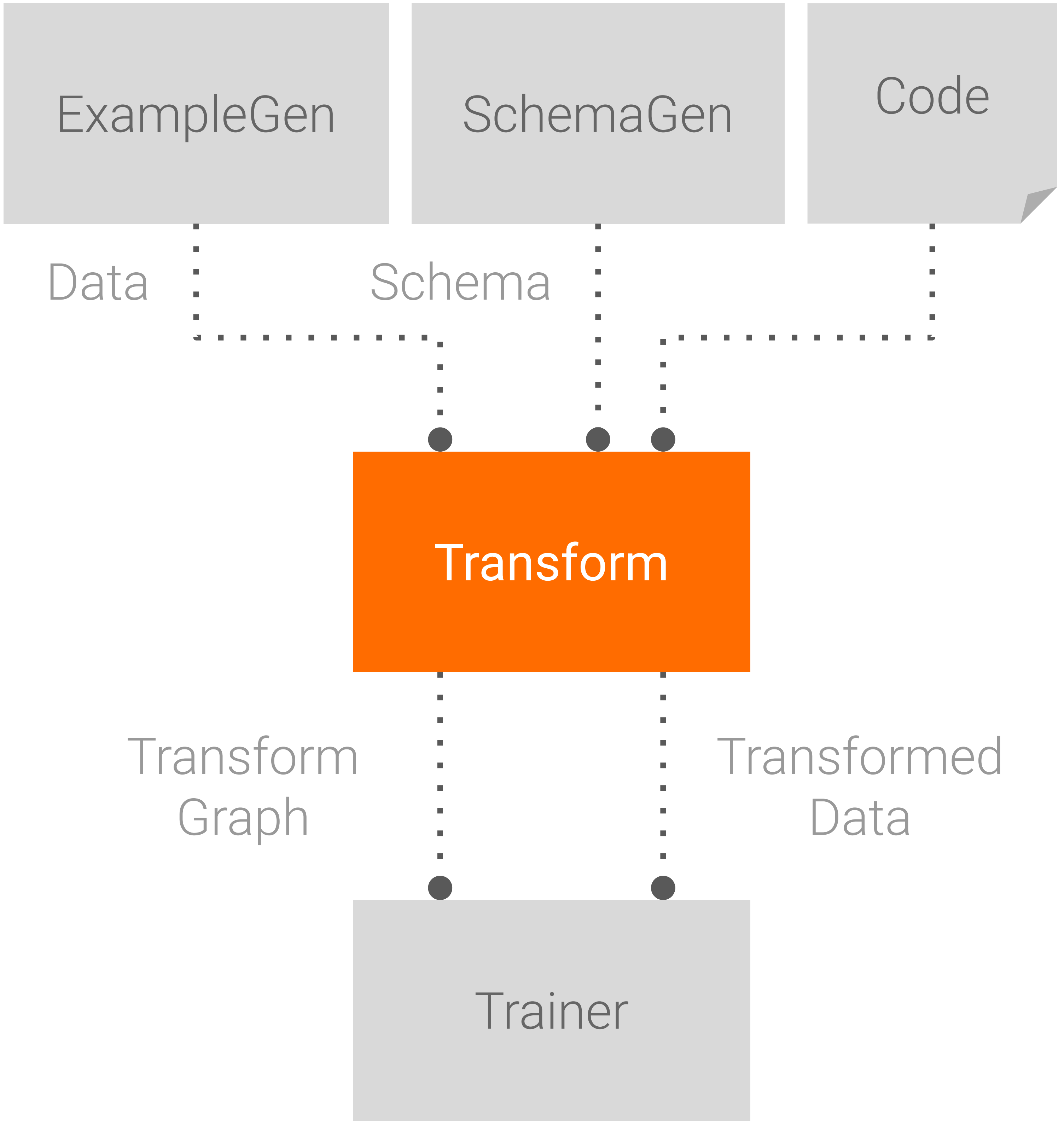
```
validate_stats = ExampleValidator(
      statistics=statistics_gen.outputs['statistics'],
      schema=infer_schema.outputs['schema'])
```

Visualization

Feature name	Anomaly short description	Anomaly long description
'payment_type'	Unexpected string values	Examples contain values missing from the schema: Prcard (<1%).
'company'	Unexpected string values	Examples contain values missing from the schema: 2092 - 61288 Sbeih company (<1%), 2192 - 73487 Zeymane Corp (<1%), 2192 - Zeymane Corp (<1%), 2823 - 73307 Seung Lee (<1%), 3094 - 24059 G.L.B. Cab Co (<1%), 3319 - CD Cab Co (<1%), 3385 - Eman Cab (<1%), 3897 - 57856 Ilie Malec (<1%), 4053 - 40193 Adwar H. Nikola (<1%), 4197 - Royal Star (<1%), 585 - 88805 Valley Cab Co (<1%), 5874 - Sergey Cab Corp. (<1%), 6057 - 24657 Richard Addo (<1%), 6574 - Babylon Express Inc. (<1%), 6742 - 83735 Tasha ride inc (<1%).

Component: Transform

Inputs and Outputs



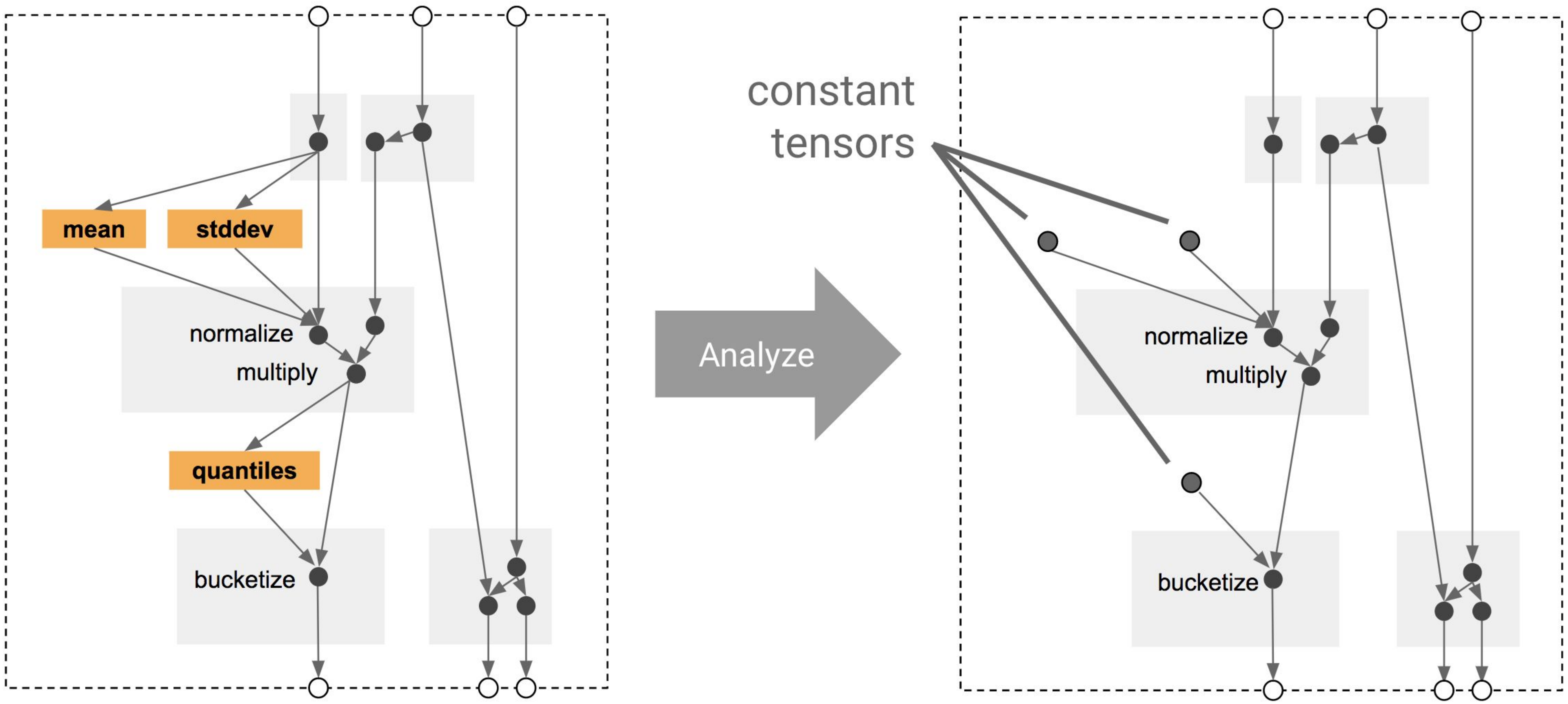
Configuration

```
transform = Transform(
      examples=example_gen.outputs['output_data'],
      schema=infer_schema.outputs['schema'],
      module_file=module_file)
```

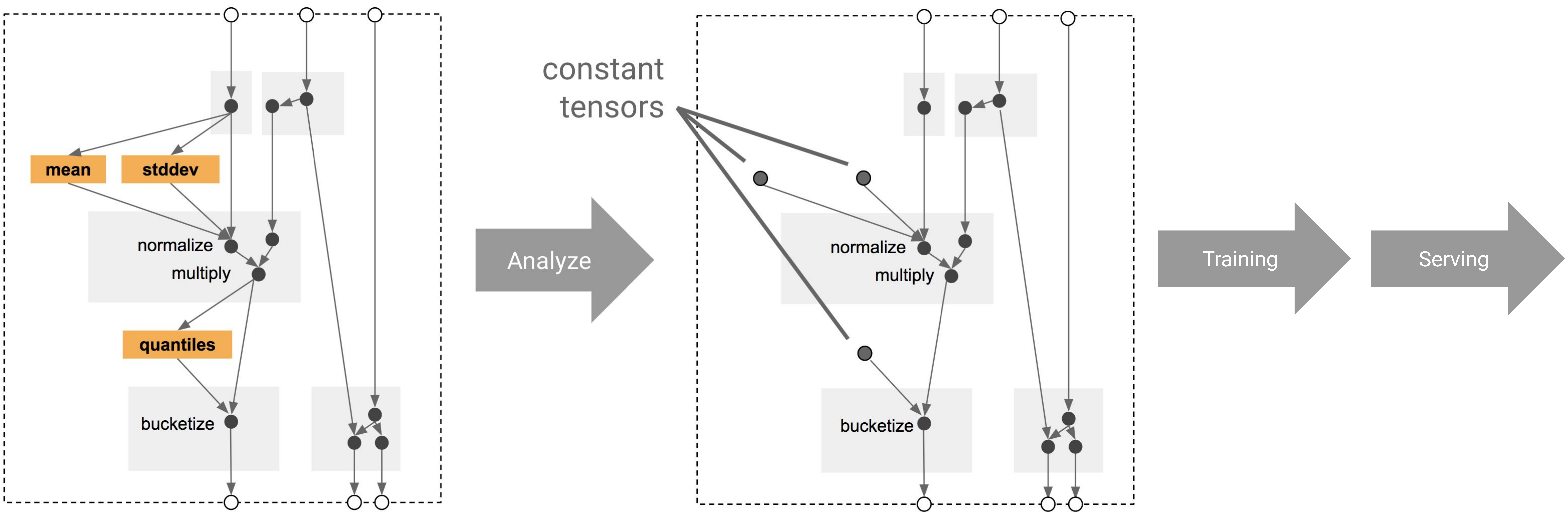
Code

```
for key in _DENSE_FLOAT_FEATURE_KEYS:
    outputs[_transformed_name(key)] = transform.scale_to_z_score(
        _fill_in_missing(inputs[key]))
# ...

outputs[_transformed_name(_LABEL_KEY)] = tf.where(
      tf.is_nan(taxi_fare),
      tf.cast(tf.zeros_like(taxi_fare), tf.int64),
      # Test if the tip was > 20% of the fare.
      tf.cast(
          tf.greater(tips, tf.multiply(taxi_fare, tf.constant(0.2))), tf.int64))

# ...
```

Using TensorFlow Transform for Feature Engineering

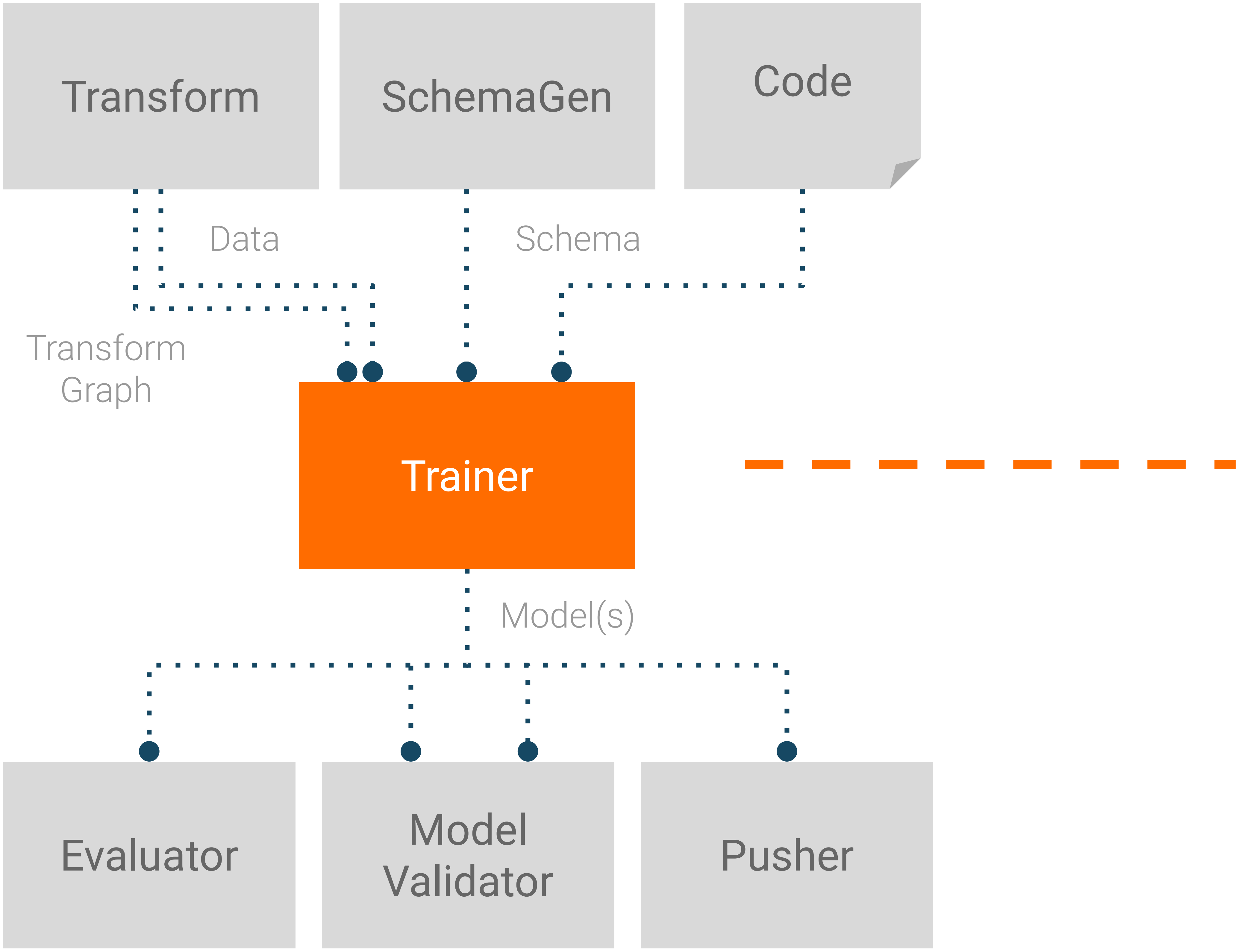


Using TensorFlow Transform for Feature Engineering



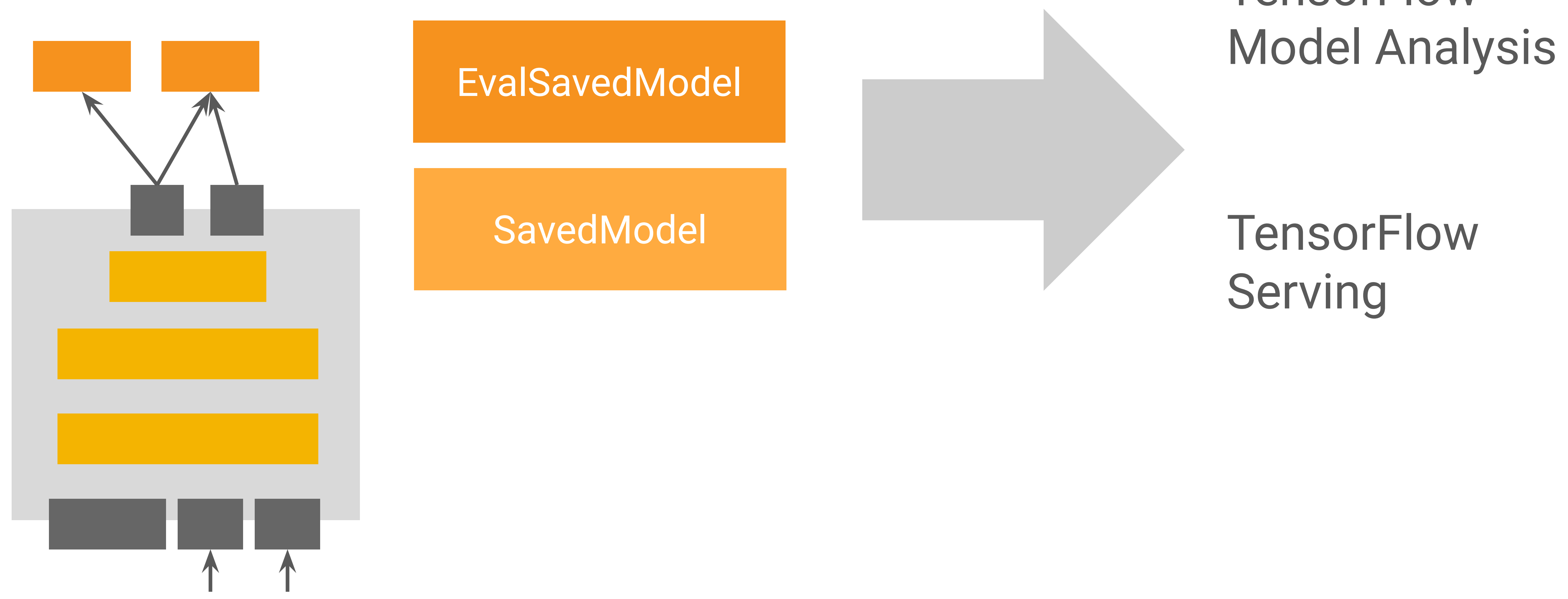
Component: Trainer

Inputs and Outputs



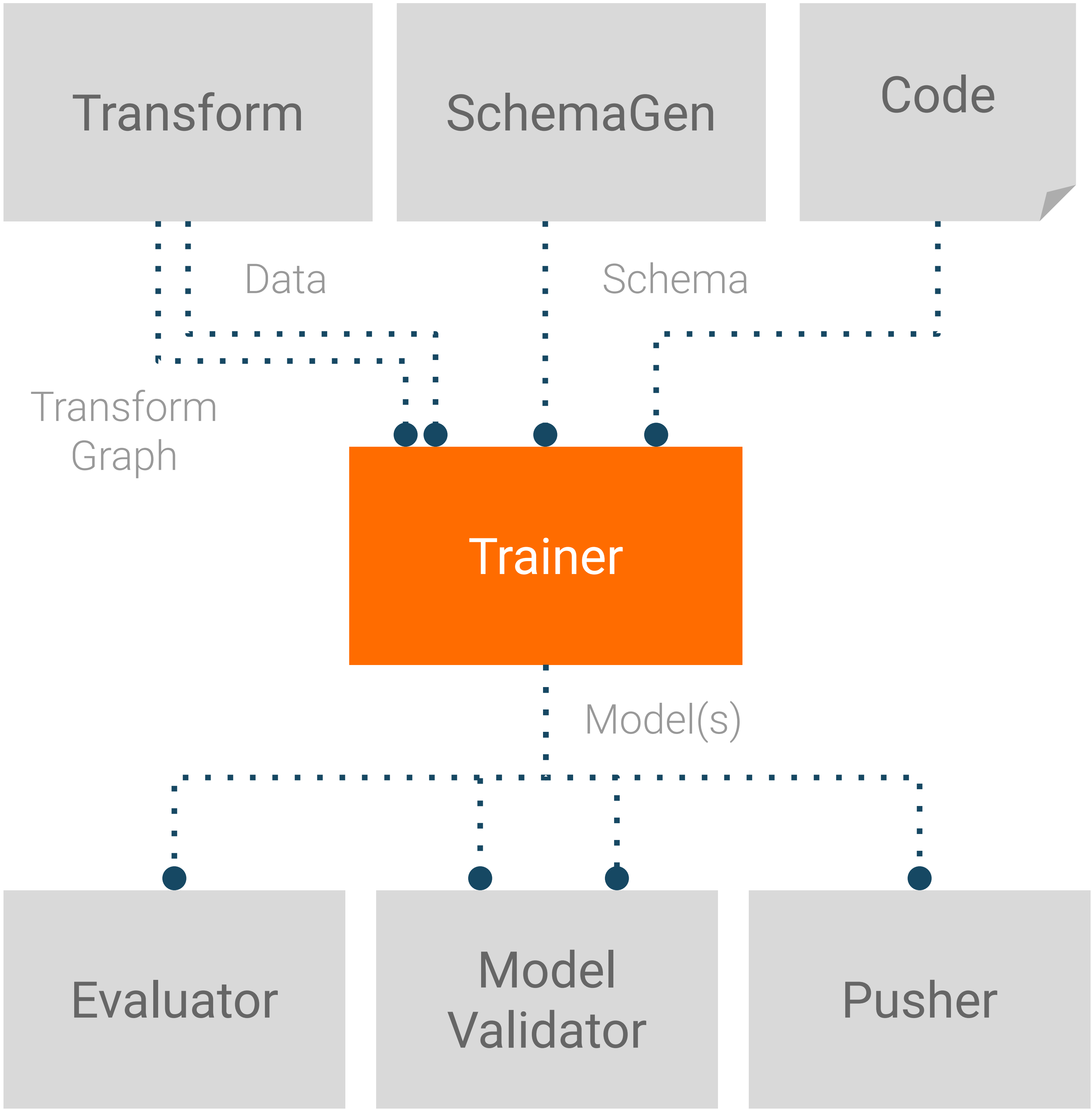
Highlight: SavedModel Format

Train, Eval, and Inference Graphs



Component: Trainer

Inputs and Outputs



Configuration

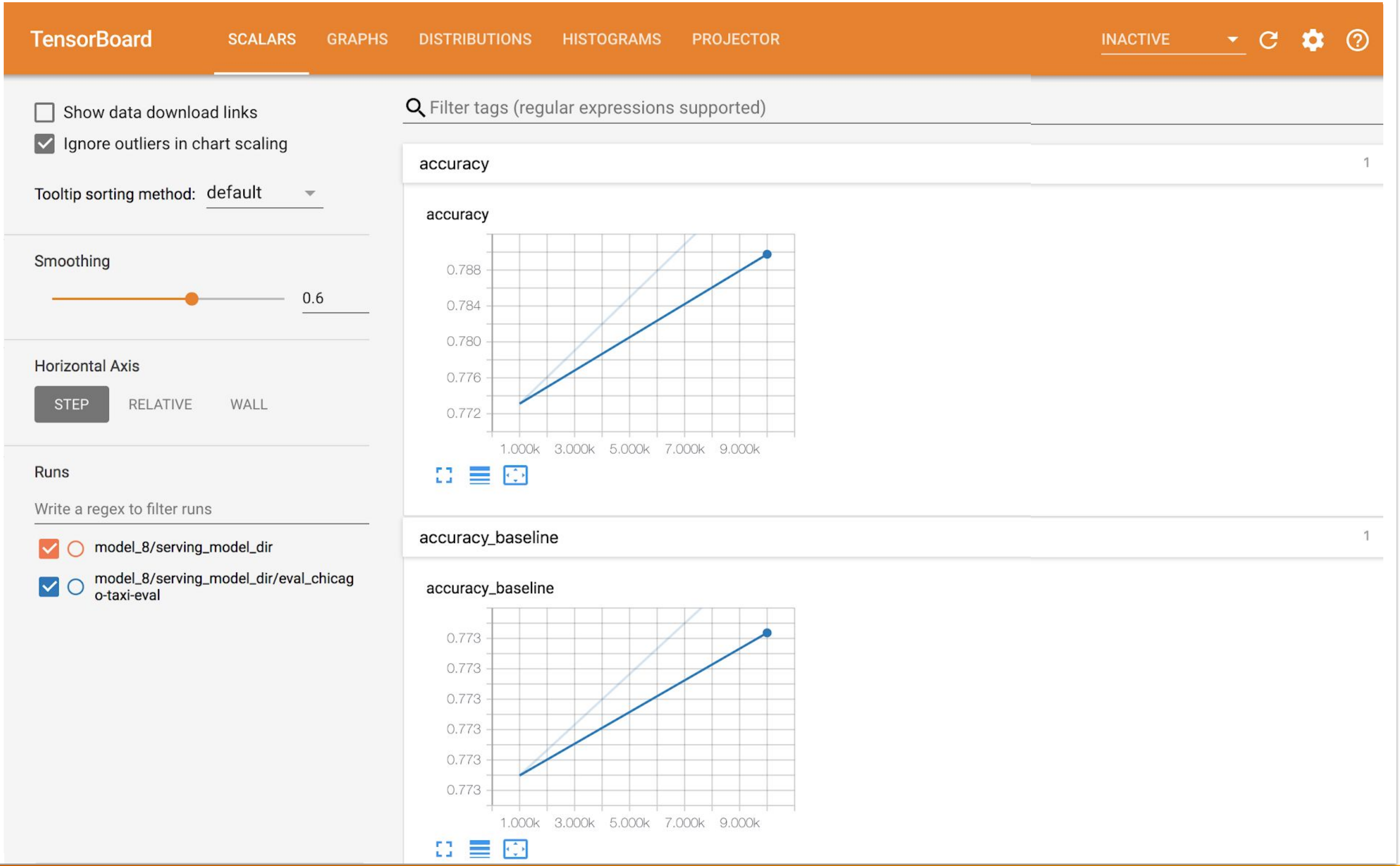
```
trainer = Trainer(
      module_file=module_file,
      transformed_examples=
          transform.outputs['transformed_examples'],
      schema=infer_schema.outputs['schema'],
      transform_graph=transform.outputs['transform_graph'],
      train_args=trainer_pb2.TrainArgs(num_steps=10000),
      eval_args=trainer_pb2.EvalArgs(num_steps=5000))
```

Code

Just TensorFlow :)

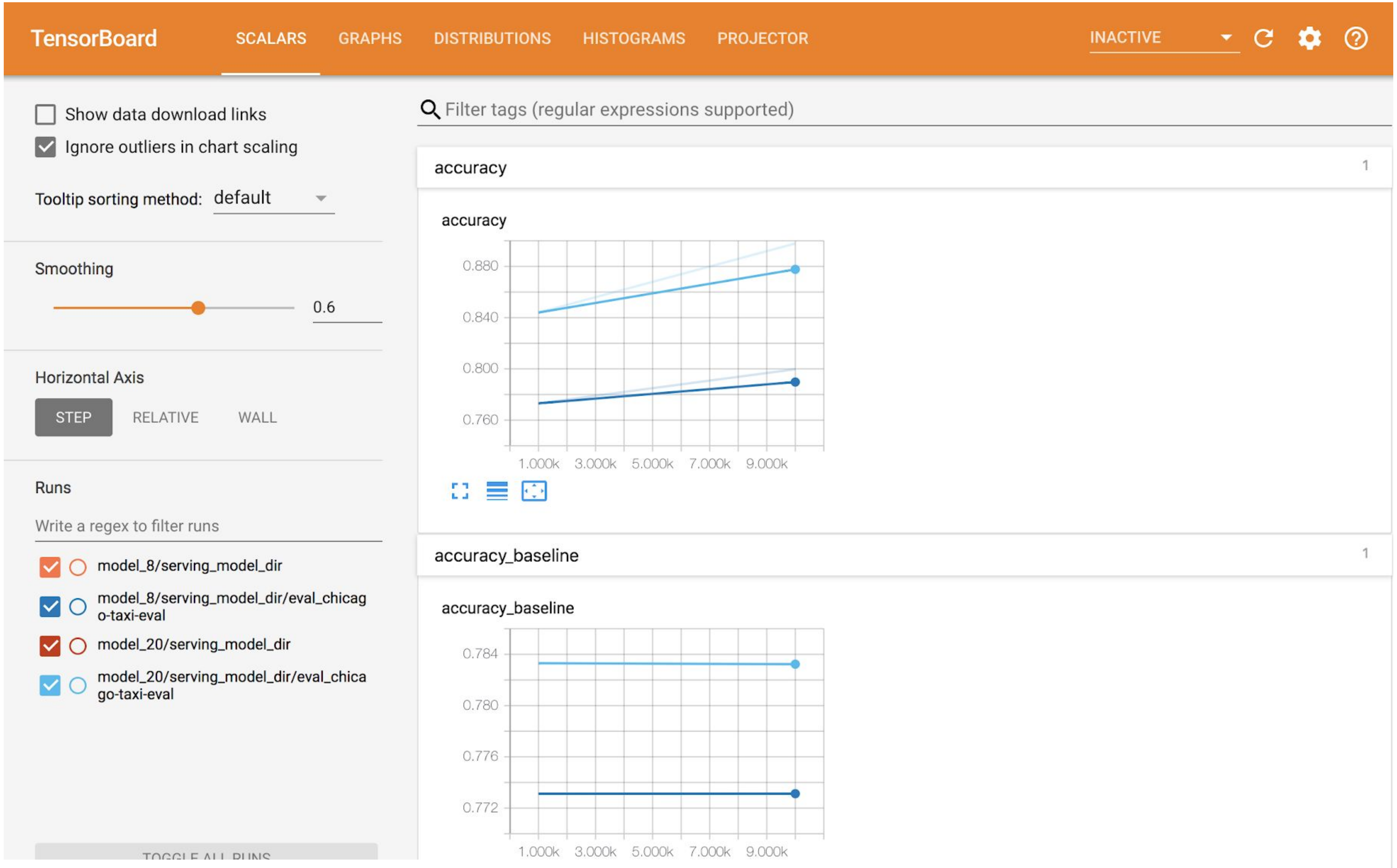
```python
# Open up Tensorboard for model_id.
print(display_tensorboard(model_id))
```

http://your.host.name:53143



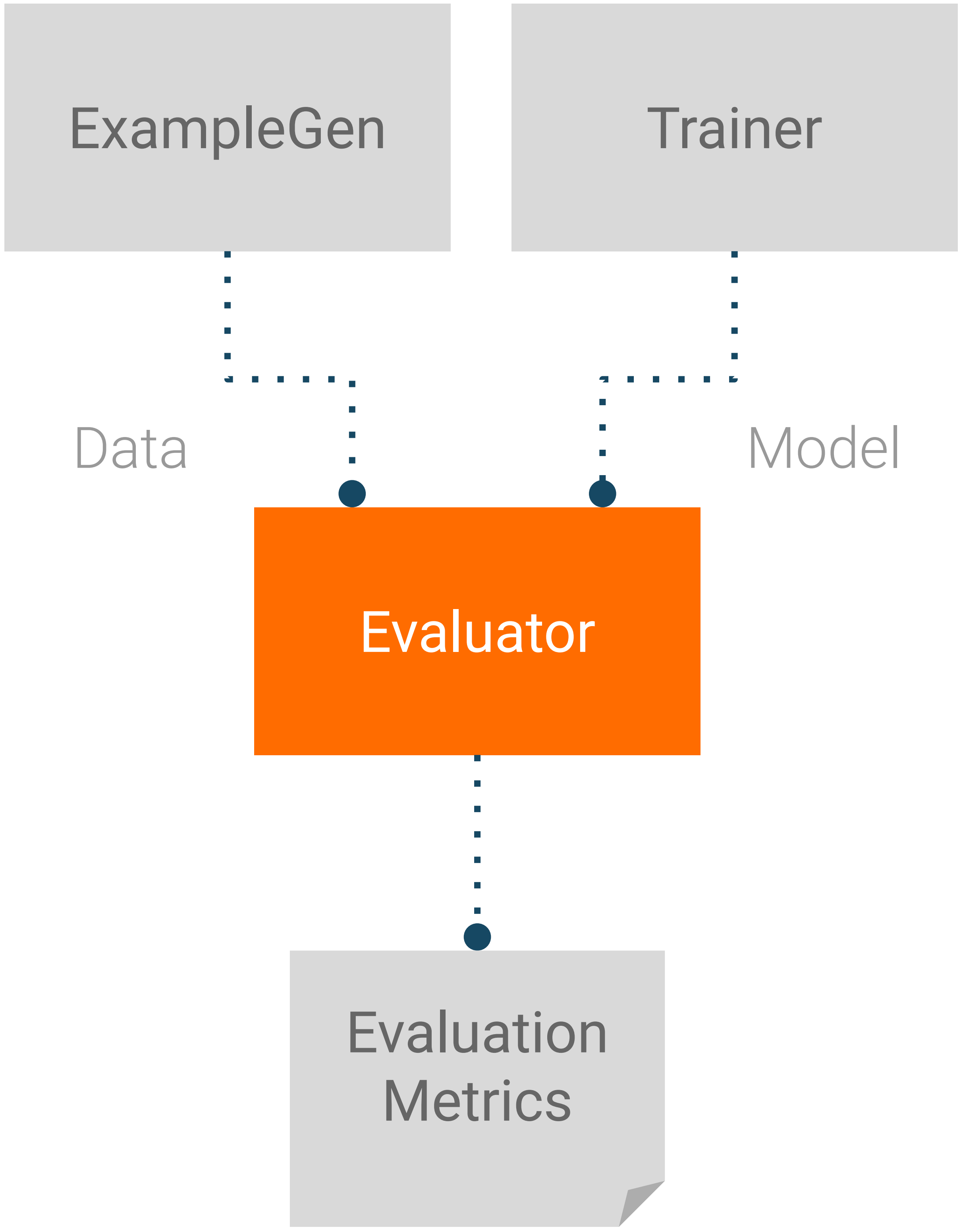
```python
# Compare Tensorboard metrics for different models.
if num_models > 1:
    print(display_tensorboard(model_id, other_model_id=other_model_id))
```

http://your.host.name:53230



Component: Evaluator

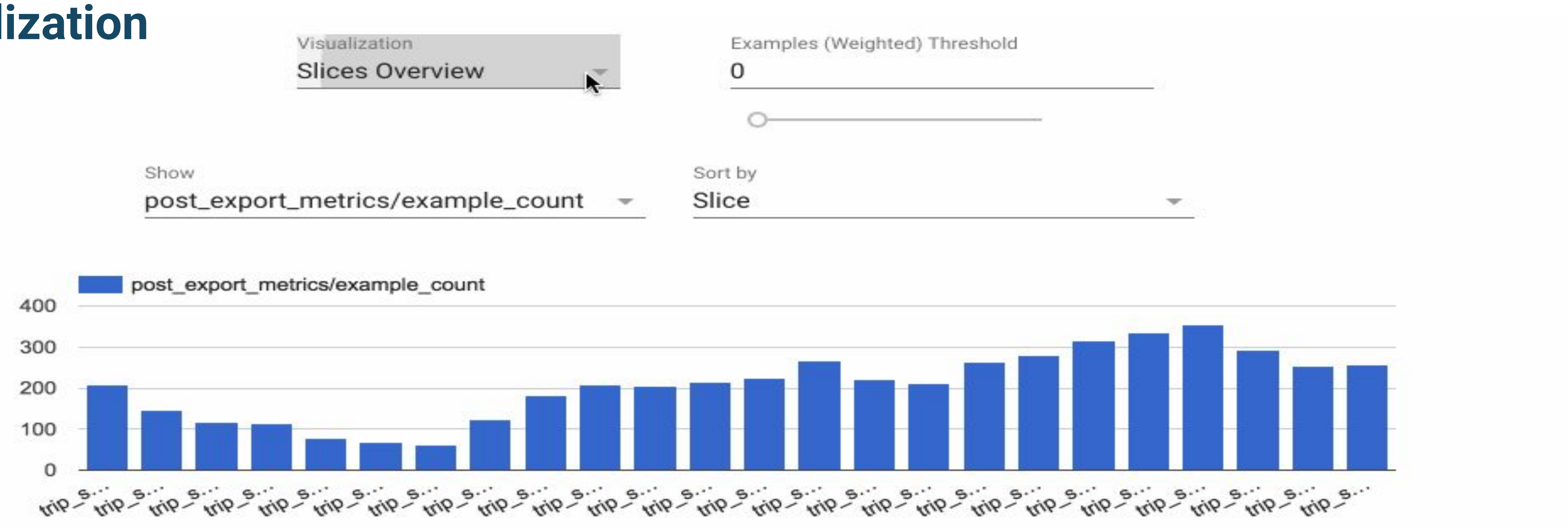
Inputs and Outputs



Configuration

```
model_analyzer = Evaluator(
      examples=example_gen.outputs['output_data'],
      model=trainer.outputs['model'],
      feature_slicing_spec=evaluator_pb2.FeatureSlicingSpec(
          specs=[evaluator_pb2.SingleSlicingSpec(
              column_for_slicing=['trip_start_hour'])]))
```

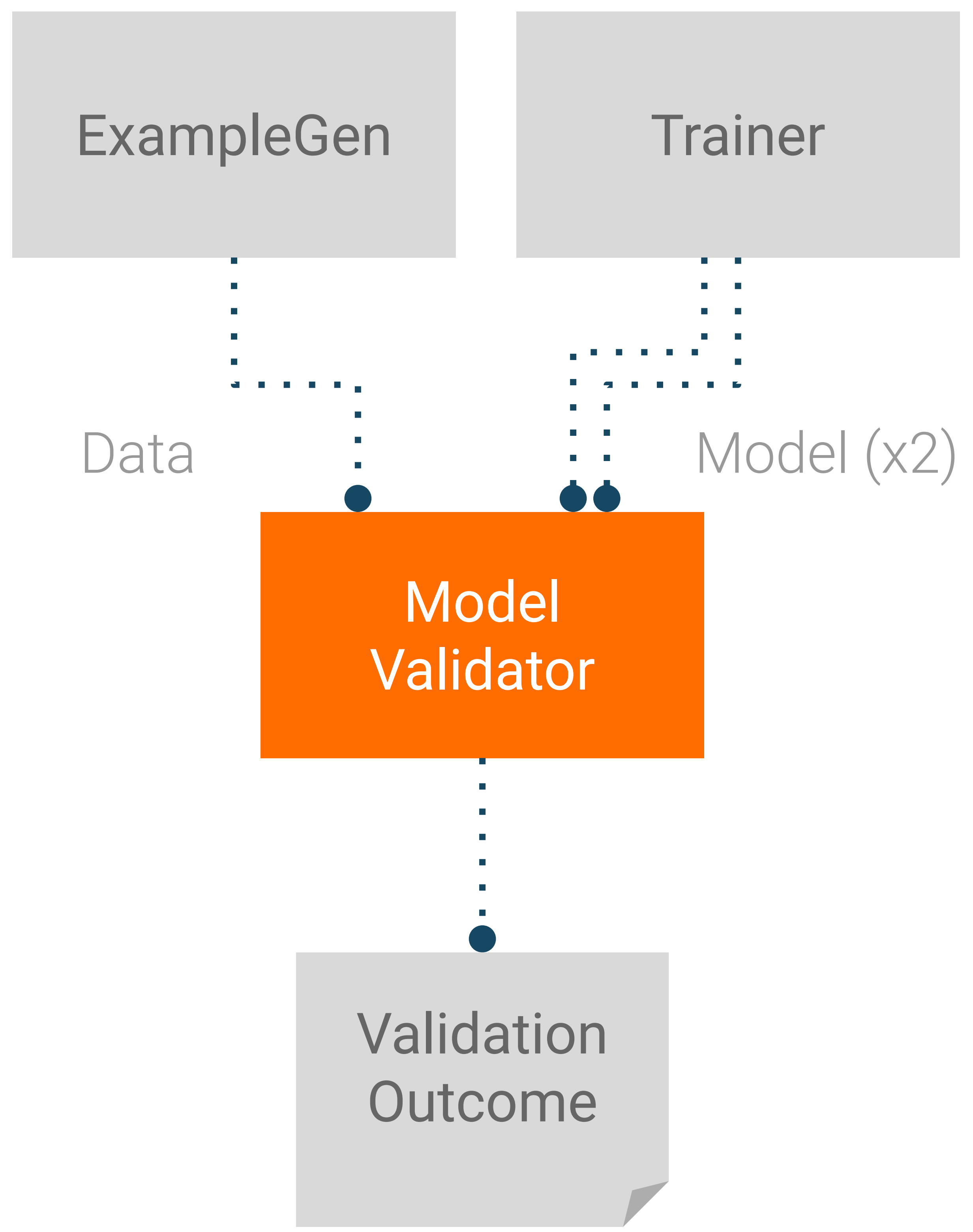
Visualization



feature	accuracy	accuracy_baseline	auc	auc_precision_recall	average_loss
trip_start_hour:19	0.63582	0.59104	0.64311	0.56092	0.64626
trip_start_hour:14	0.67117	0.65766	0.63793	0.49112	0.61667
trip_start_hour:2	0.66102	0.63559	0.58527	0.47002	0.65236
trip_start_hour:12	0.69643	0.65625	0.68270	0.54122	0.59538
trip_start_hour:0	0.66184	0.66667	0.63773	0.45081	0.61634
trip_start_hour:23	0.65625	0.64844	0.58357	0.43514	0.64315

Component: ModelValidator

Inputs and Outputs



Configuration

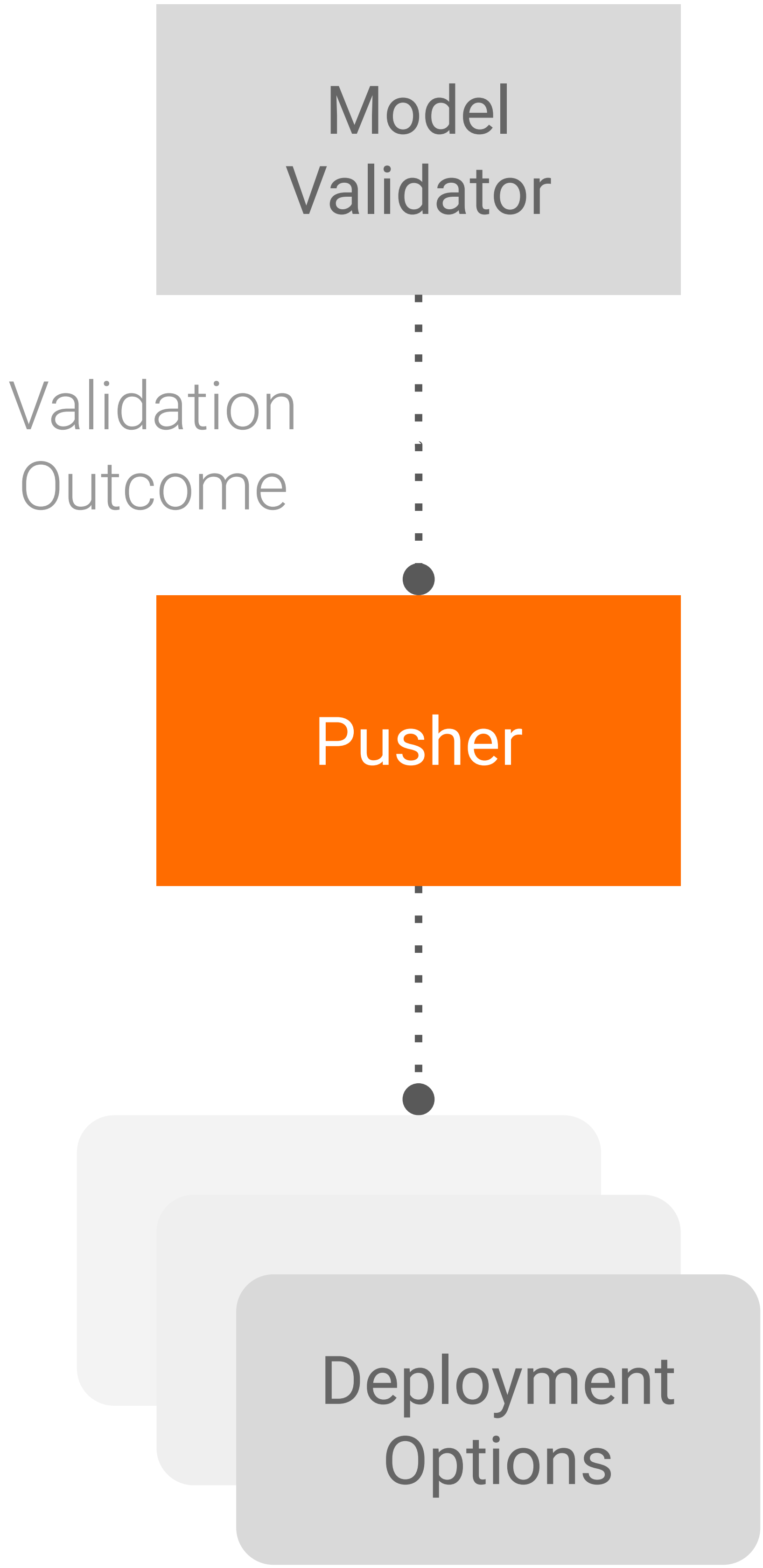
```
model_validator = ModelValidator(
      examples=example_gen.outputs['output_data'],
      model=trainer.outputs['model'])
```

Configuration Options

- Validate using current eval data
- “Next-day eval”, validate using unseen data

Component: Pusher

Inputs and Outputs



Configuration

```
pusher = Pusher(
      model=trainer.outputs['model'],
      model_blessing=model_validator.outputs['blessing'],
      push_destination=pusher_pb2.PushDestination(
          filesystem=pusher_pb2.PushDestination.Filesystem(
              base_directory=serving_model_dir)))
```

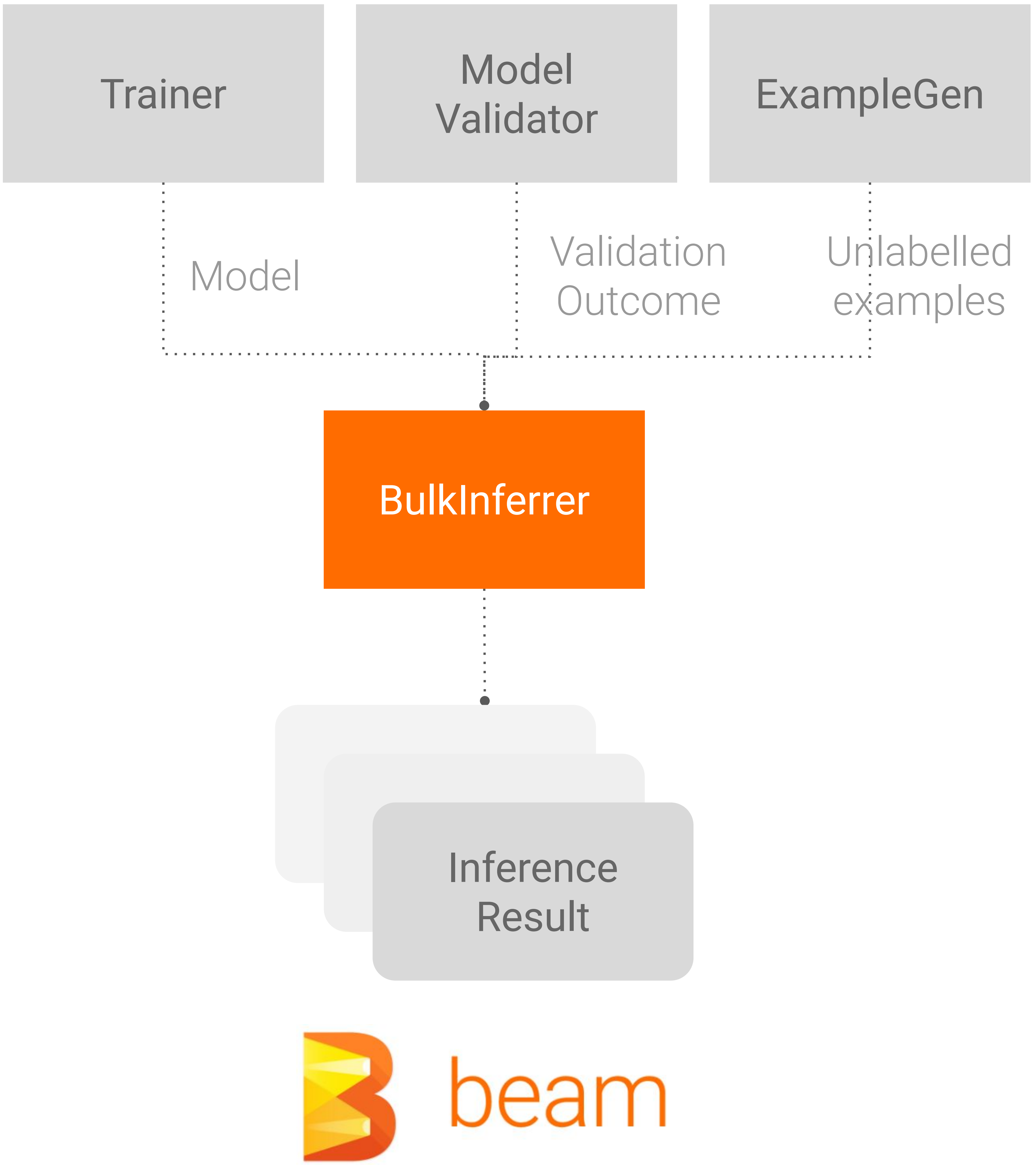
Block push on validation outcome

Push destinations supported today

- Filesystem (TensorFlow Lite, TensorFlow JS)
- TensorFlow Serving

Component: BulkInferrer

Inputs and Outputs



Configuration

```
bulk_inferrer = BulkInferrer(
    examples=inference_example_gen.outputs['examples'],
    model_export=trainer.outputs['output'],
    model_blessing=model_validator.outputs['blessing'],
    data_spec=bulk_inferrer_pb2.DataSpec(
        example_splits=['unlabelled']),
    model_spec=bulk_inferrer_pb2.ModelSpec())
```

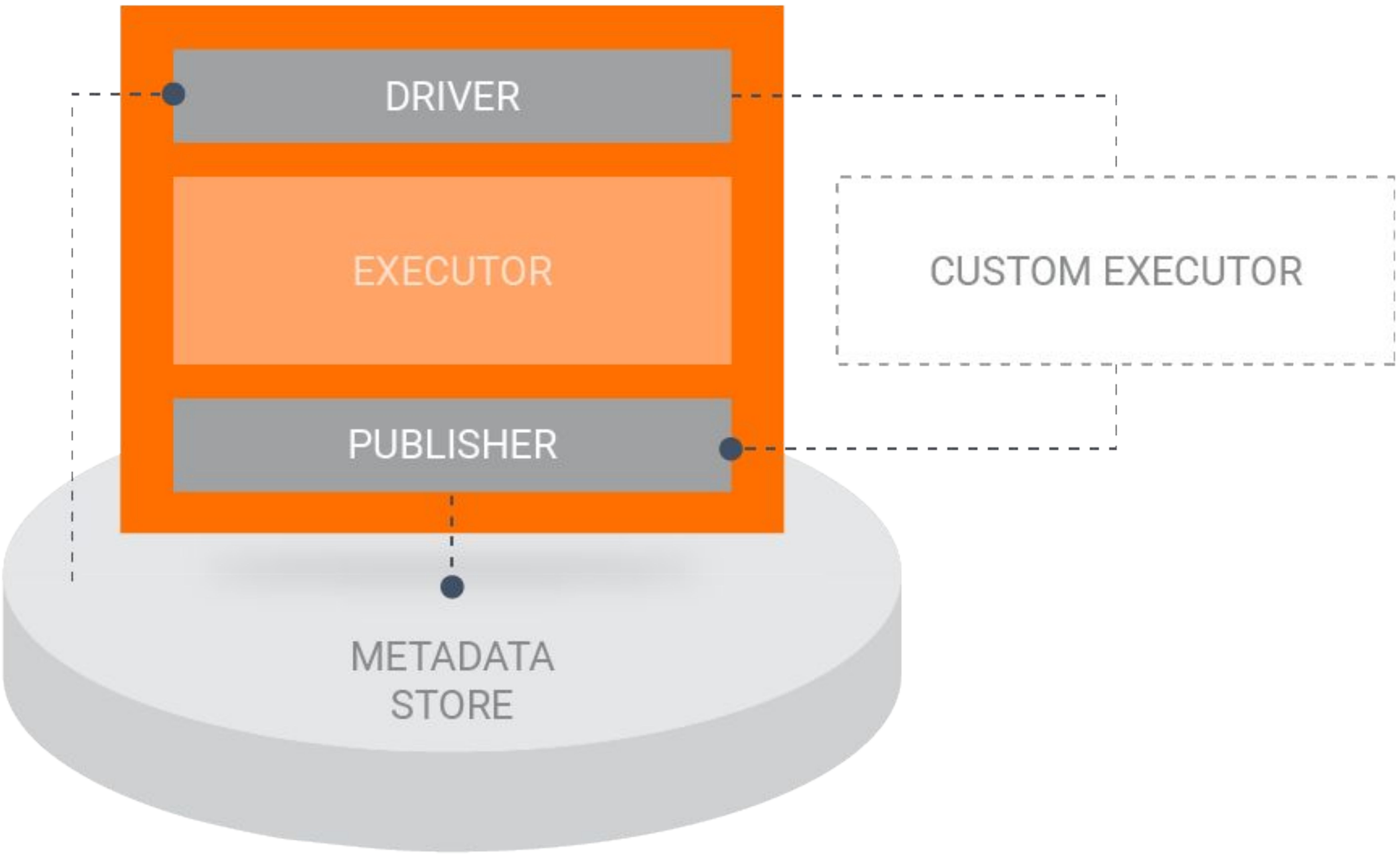
Configuration Options

Block batch inference on a successful model validation.
Choose the inference examples from example gen's output.
Choose the signatures and tags of inference model.

Inference Result

Contains features and predictions.

TFX Custom Components



Extend the existing components

Replace the default component executor with your own code, providing the ability to extend existing components with your own implementation.

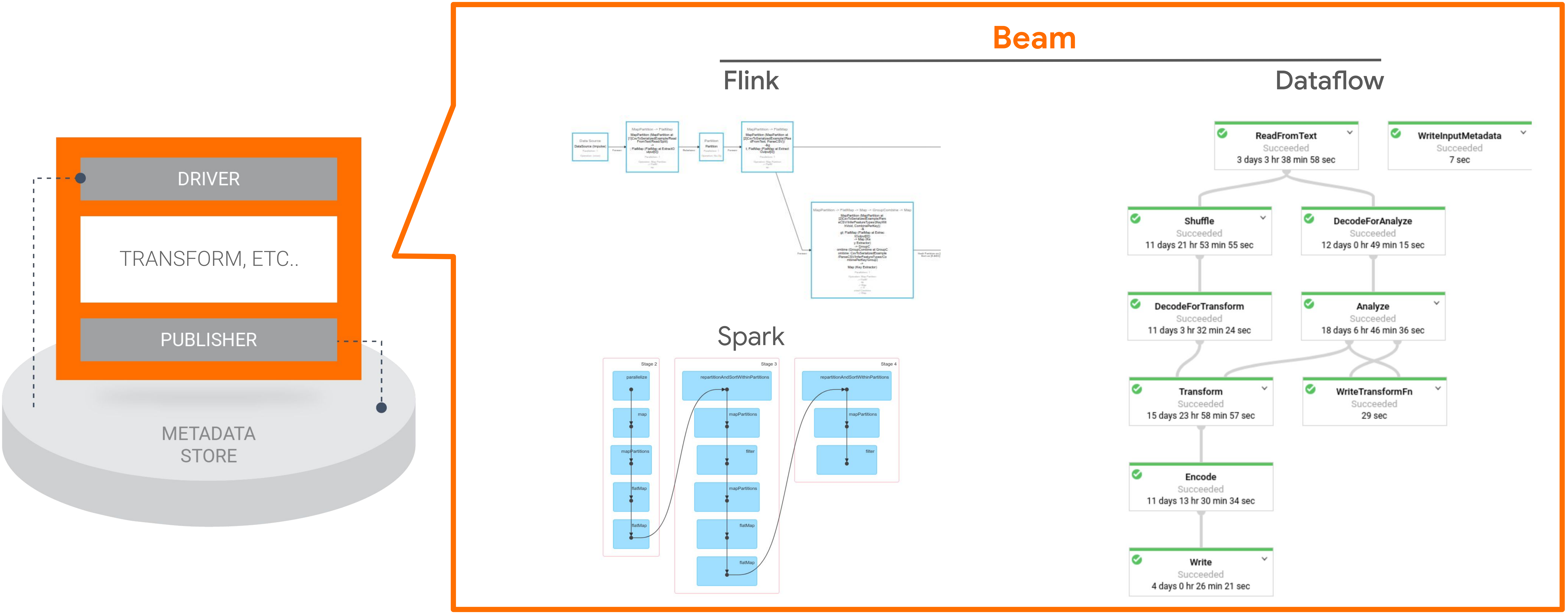
Semi-Custom Component:

Overriding with your own executor

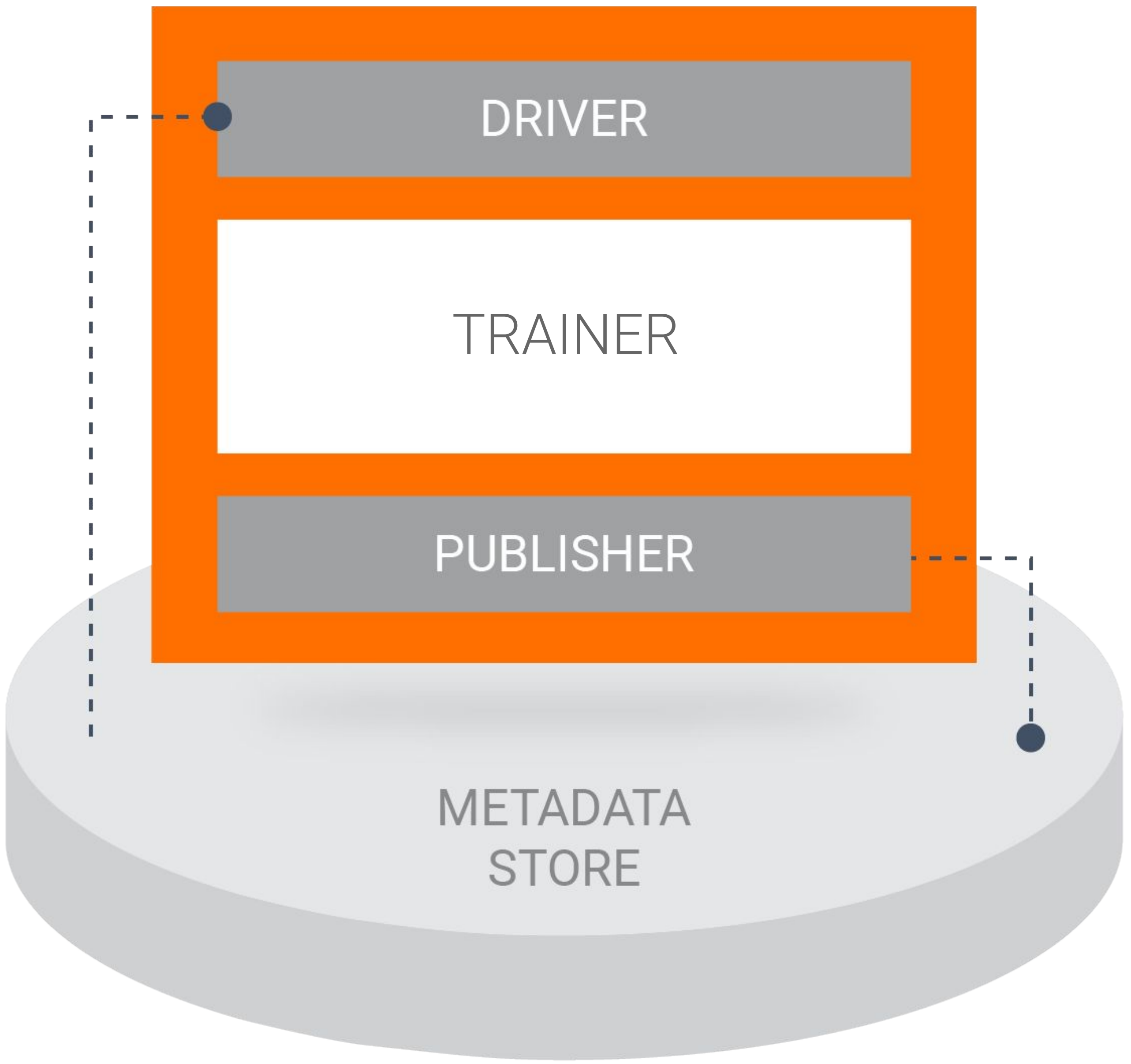
- Start with an existing component
- Extend **BaseExecutor** and implement **Do()**
- Add your custom code into the Do() method
- Use custom_config parameters to add inputs to the custom executor

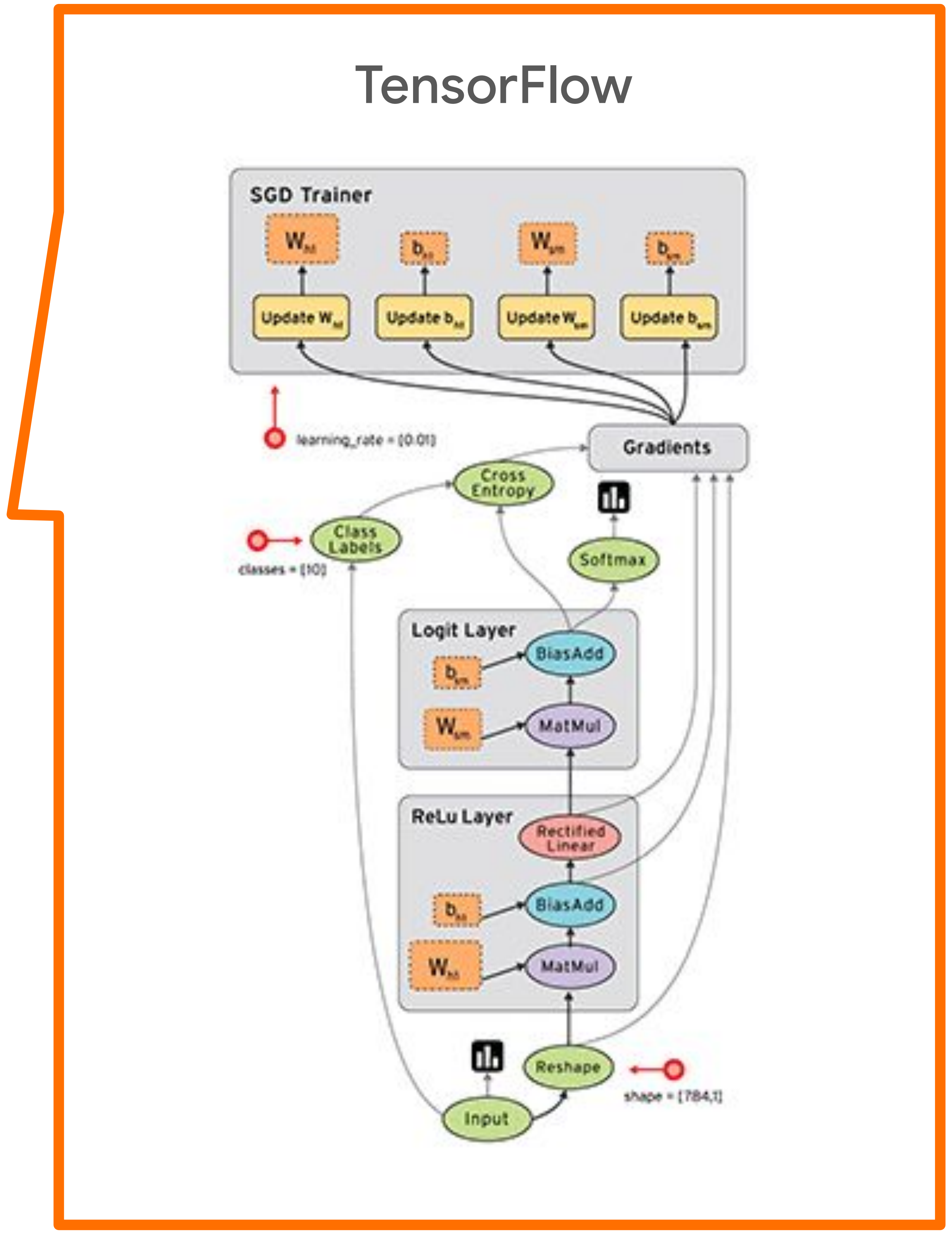
```
class Executor(base_executor.BaseExecutor):

  def Do(self, input_dict,
            output_dict,
            exec_properties):
```

Executors do the work

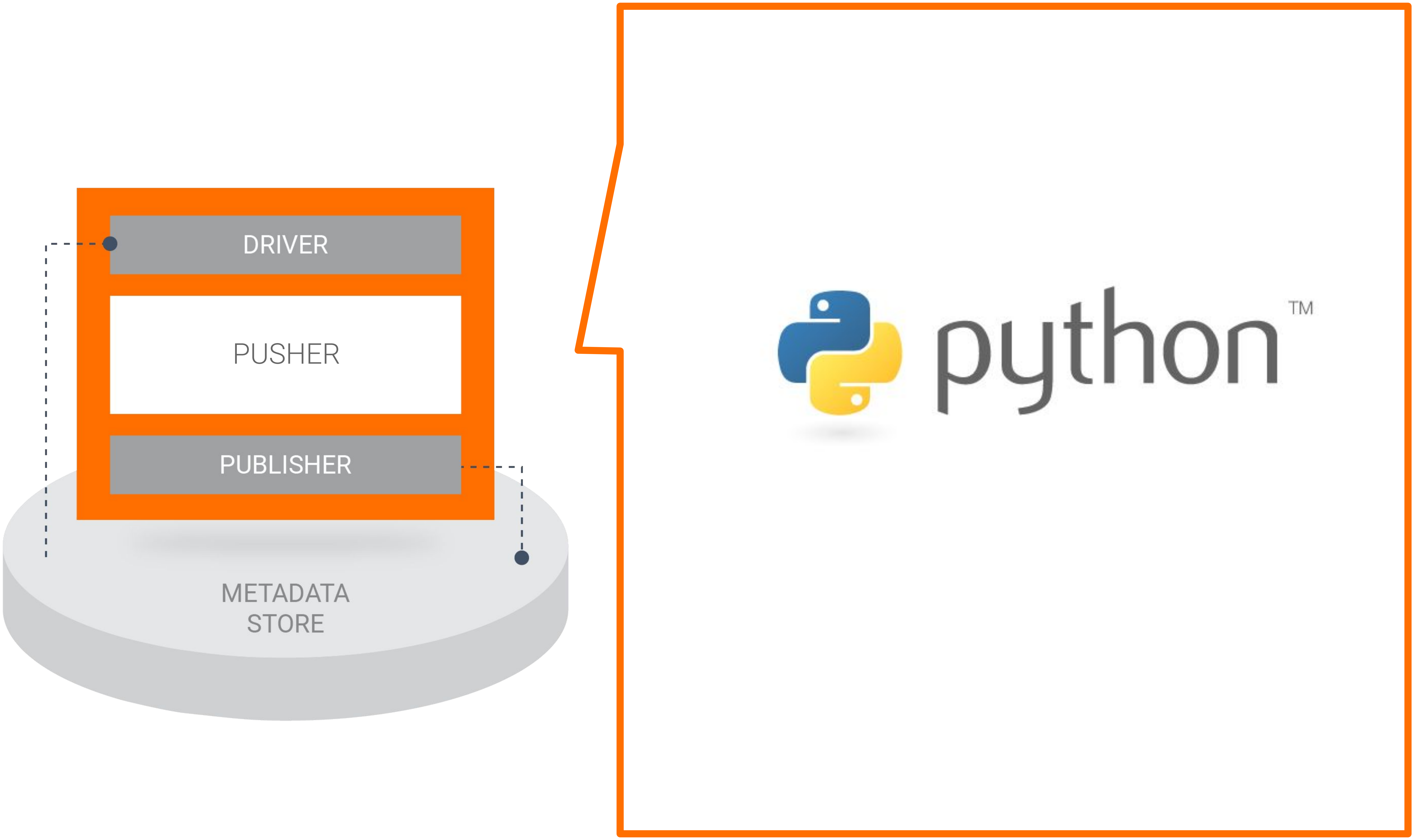


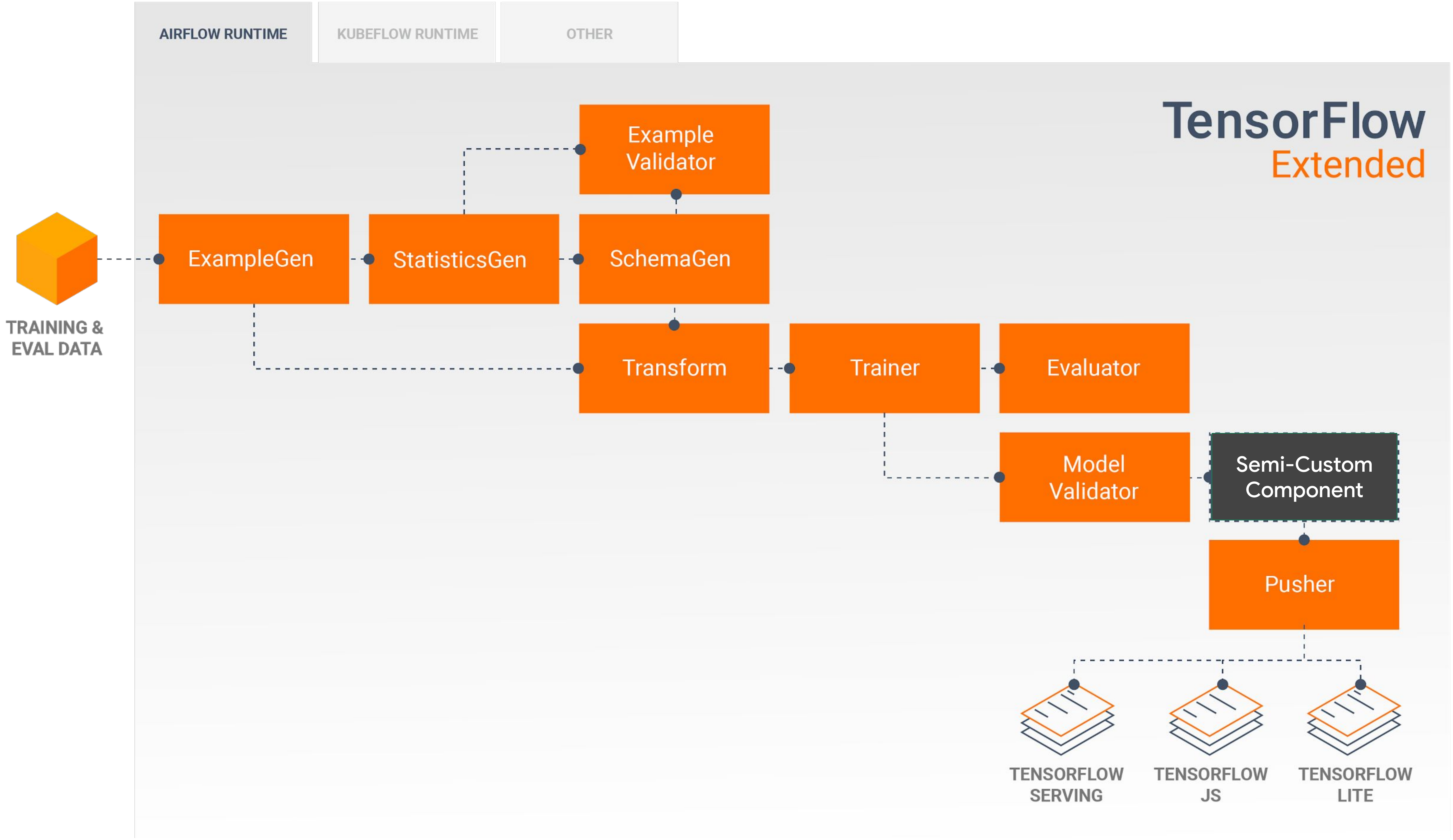
Executors do the work

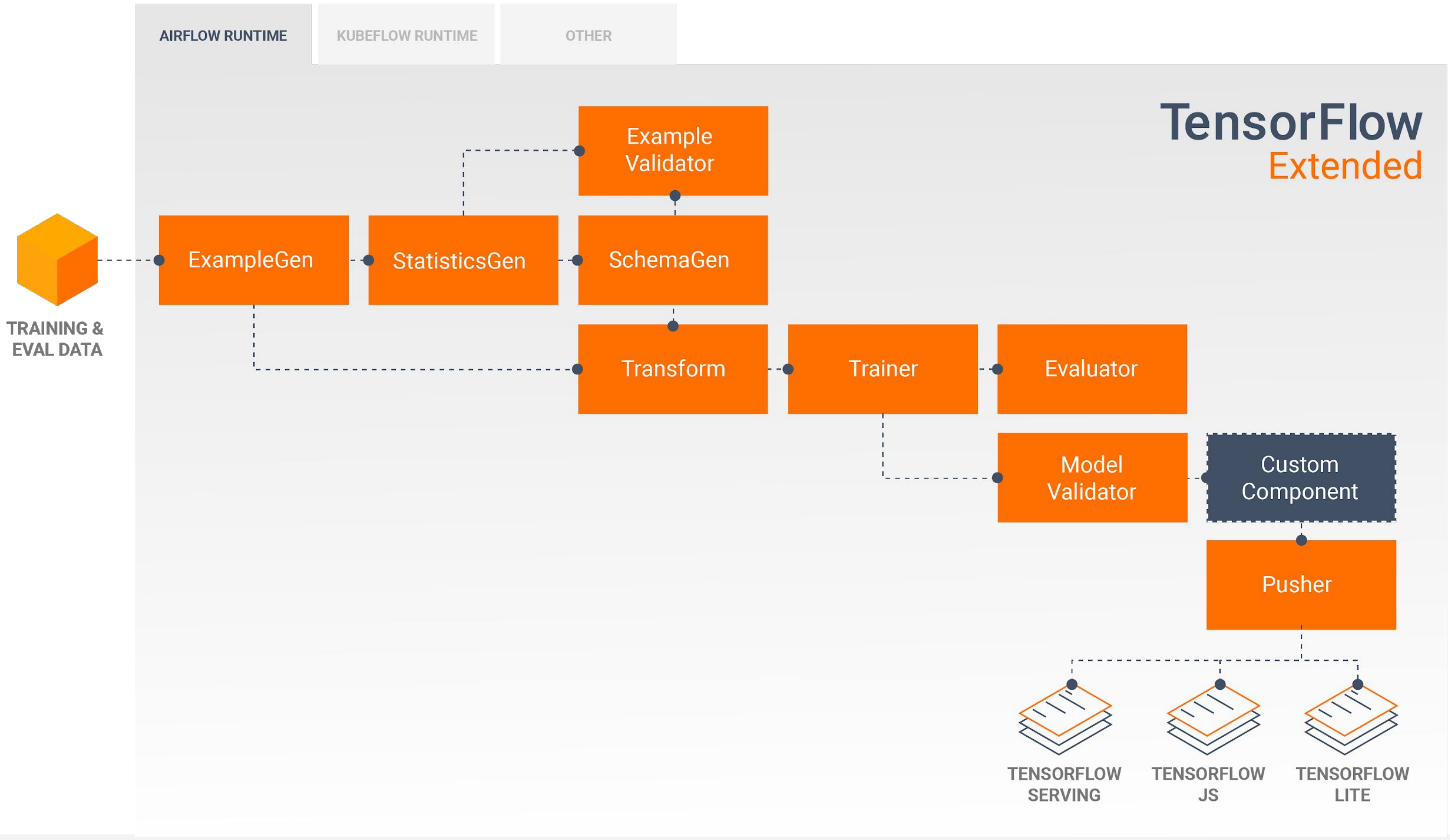




Executors do the work




AIRFLOW RUNTIME
KUBEFLOW RUNTIME
OTHER
TensorFlow
Extended
TRAINING &
EVAL DATA
ExampleGen
StatisticsGen
Example Validator
SchemaGen
Transform
Trainer
Evaluator
Model Validator
Semi-Custom Component
Pusher
TENSORFLOW SERVING
TENSORFLOW JS
TENSORFLOW LITE



Build your own component

Create your own components to run within a TFX pipeline while still providing the benefits of metadata management, lineage, and pipeline monitoring.

Custom Component:

New Component Inputs & Outputs

Use **`ComponentSpec`** to define the new inputs and outputs

- **INPUTS**: Input artifacts that will be passed into the executor
- **OUTPUTS:** Output artifacts which the executor will produce
- **PARAMETERS:** Additional properties required by the executor. These are non-artifact parameters defined in the pipeline DSL and passed into execution.

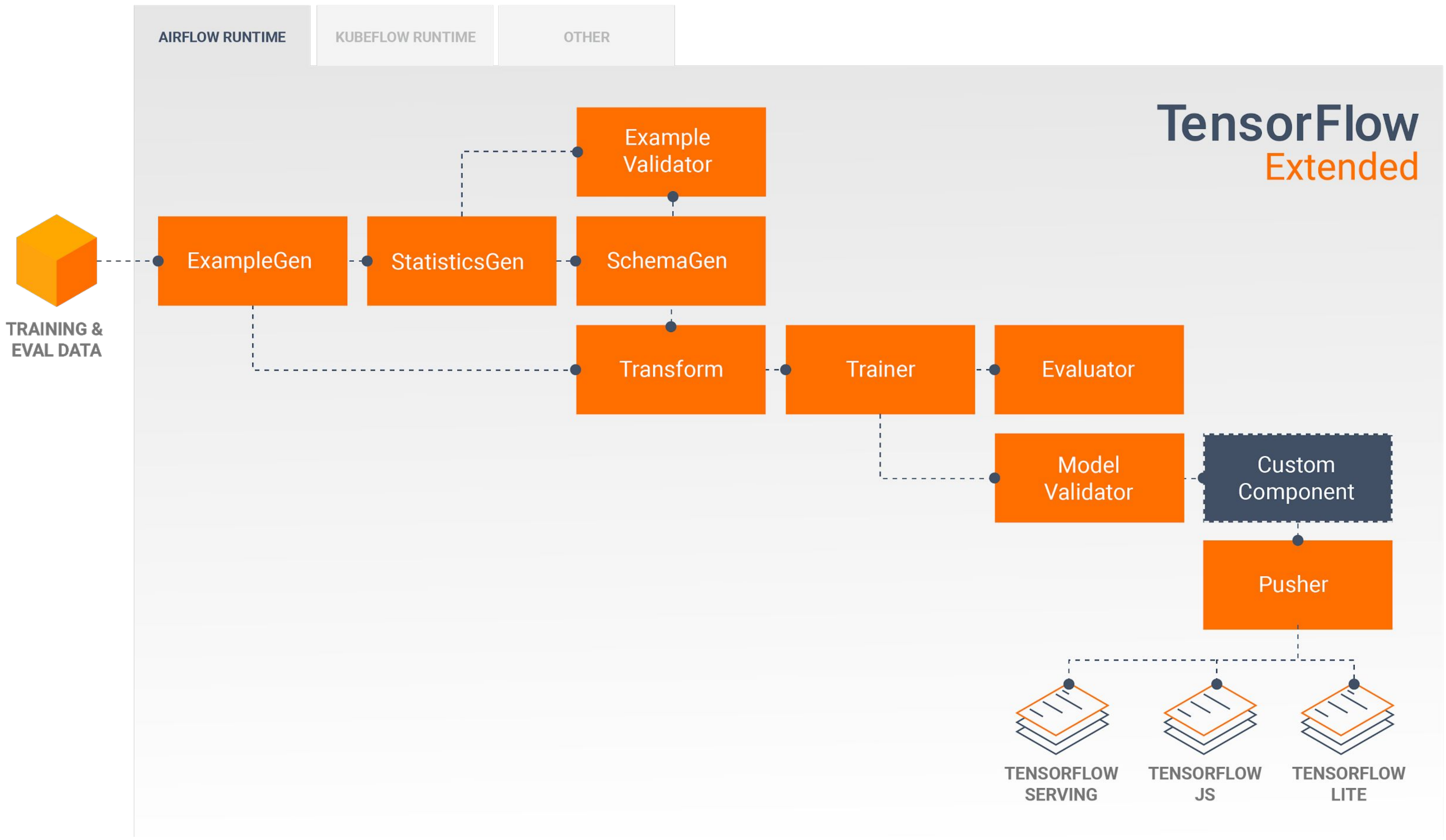
```
class MyComponentSpec(types.ComponentSpec):

  PARAMETERS = {
     'timeout_sec': ExecutionParameter(type=int),
  }
  INPUTS = {
     'model_export': ChannelParameter(type_name='ModelExportPath'),
  }
  OUTPUTS = {
     'MyBlessing': ChannelParameter(type_name='ModelBlessingPath'),
  }
```

Custom Component:

Your Executor, Inputs, Outputs, and Params

Same as when just doing a custom executor, but with the custom inputs and outputs defined in your custom ComponentSpec

```
class Executor(base_executor.BaseExecutor):
  """Start a trainer job on Google Cloud AI Platform."""

  def Do(self, input_dict,
         output_dict,
         exec_properties):
    """Starts a trainer job on Google Cloud AI Platform.
```



AIRFLOW RUNTIME
KUBEFLOW RUNTIME
OTHER
TensorFlow
Extended
TRAINING &
EVAL DATA
ExampleGen
StatisticsGen
Example Validator
SchemaGen
Transform
Trainer
Evaluator
Model Validator
Custom Component
Pusher
TENSORFLOW SERVING
TENSORFLOW JS
TENSORFLOW LITE

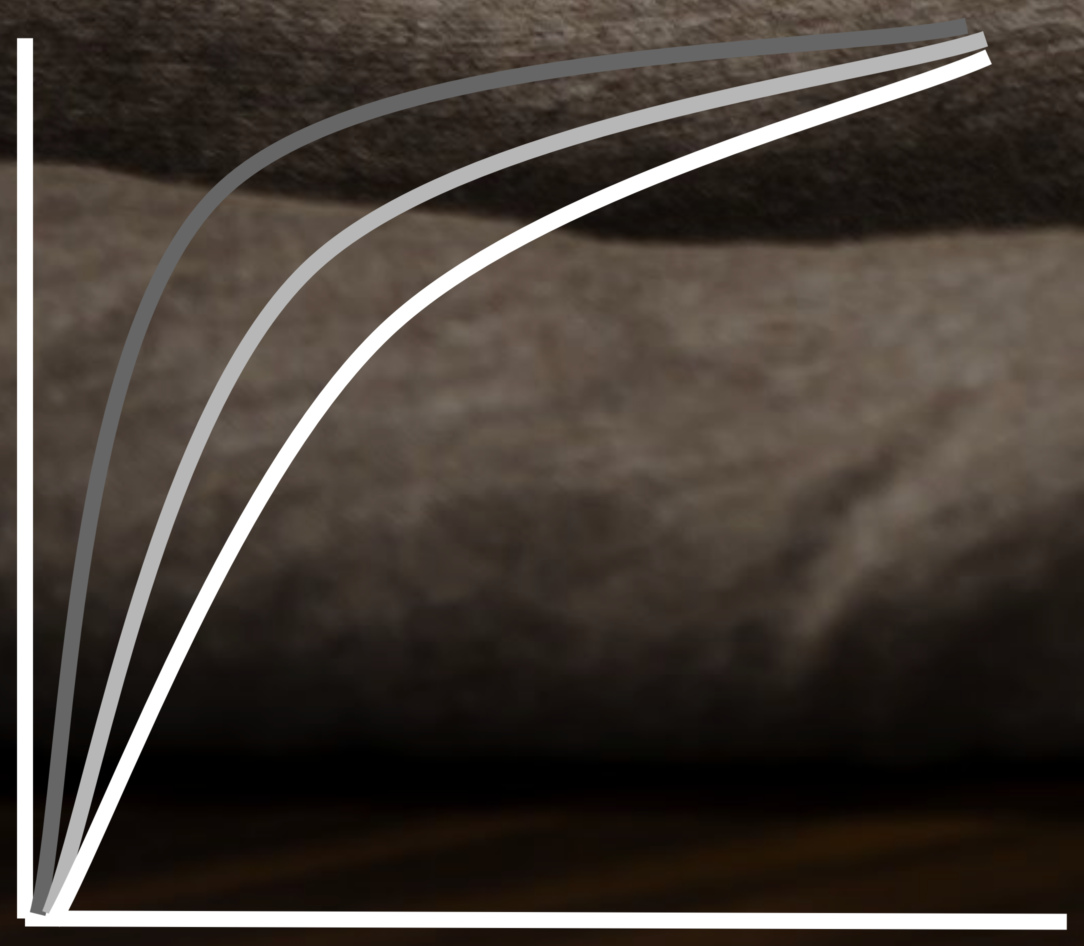
... back to the shoes

You’re an Online Retailer Selling Shoes ...

Your model predicts **click-through rates (CTR)**, helping you decide how much inventory to order

When suddenly

Your AUC and prediction accuracy have **dropped** on men’s dress shoes!

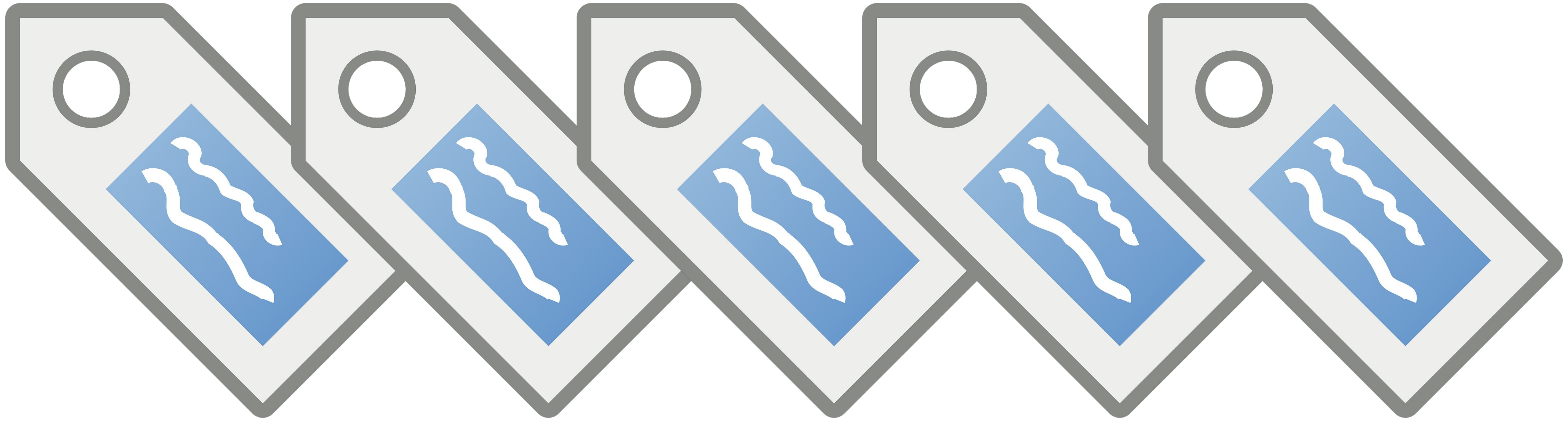


Detecting Problems With Deployed Models

- Problems are with current inference results
  - Example data will be from **current inference requests**
  - **Not your training data**
- Monitor to find problems early

Detecting Problems With Deployed Models

- To measure model performance, you need labels
  - Process feedback - Example: Actual versus predicted click-through
  - Semi-supervision - Human labeling - Expensive, limited
  - Weak supervision - Historical data, heuristics, comparison to other models



First Things First

Check your data with the ExampleValidator component and the tools in TensorFlow Data Validation:

- No outliers
- No missing features
- Minimal distribution shift

Sort by

Feature order

Reverse order

Feature search

Features: int(5) float(10) string(2) unknown(1)

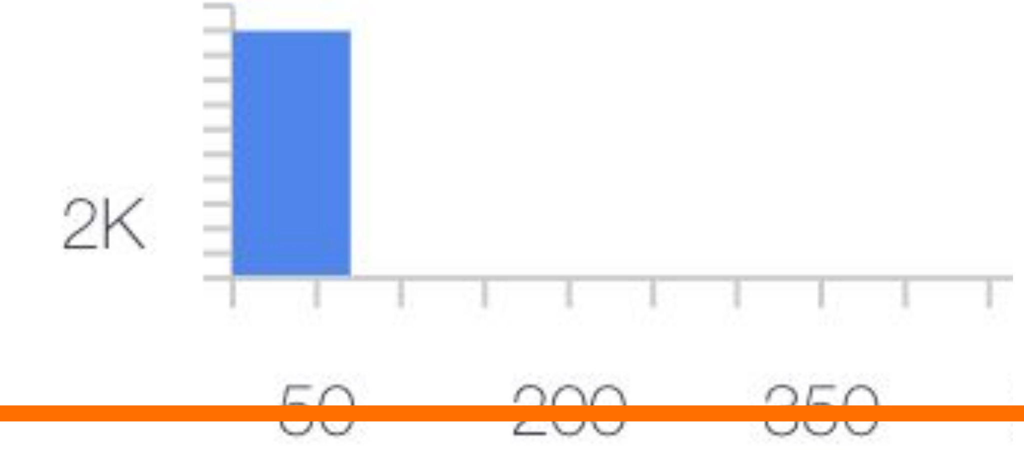
Numeric Features (15)

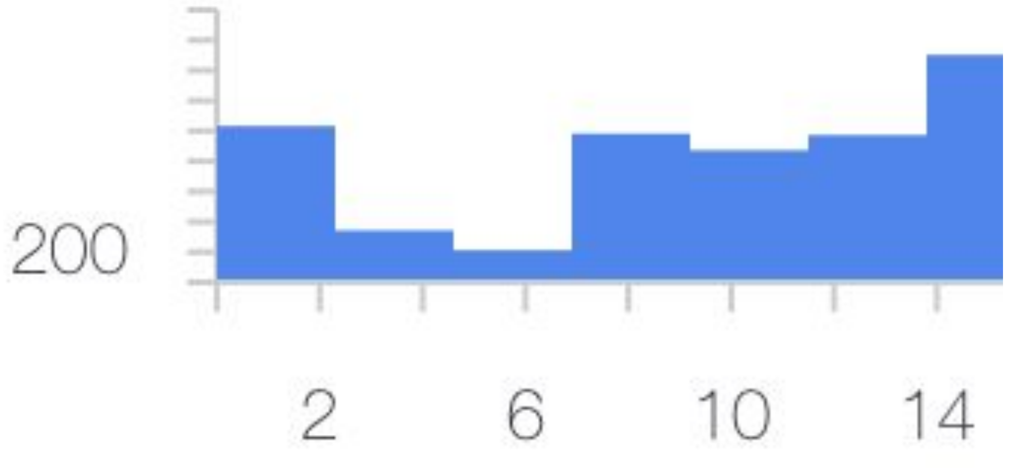
Chart to show

Standard

log expand

	count	missing	mean	std dev	zeros	min	median	max
price	10,000	0%	11.74	12.13	0.17%	0	7.85	700.07
shoe_size	10,000	0%	13.63	6.61	4.14%	0	15	23





Analyze your model performance

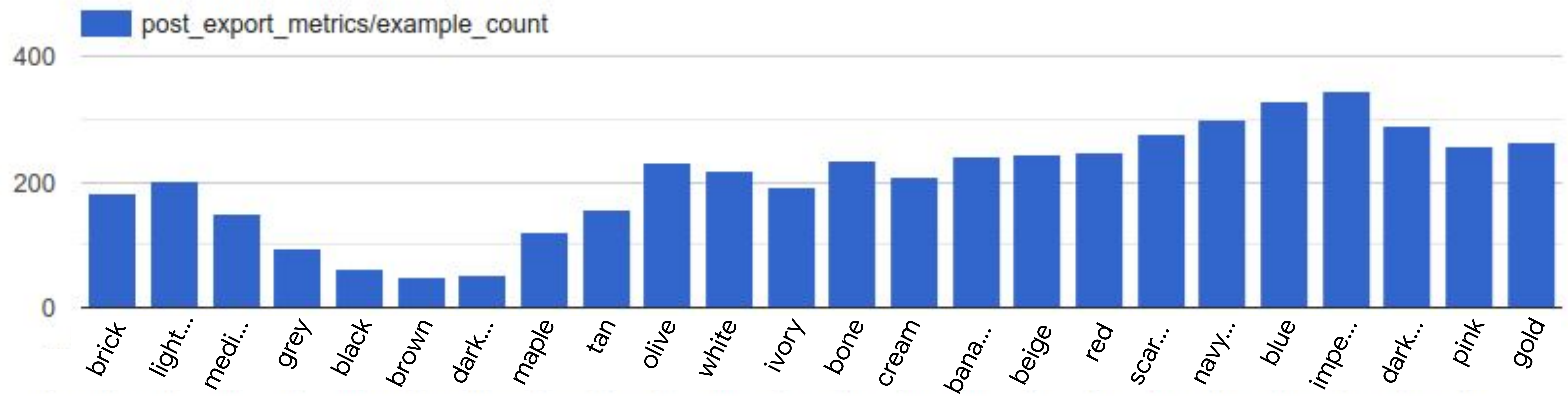
- **Check your current model performance with current data**
  - Define slices for your domain - like Men's Dress Shoes
  - Create labeled dataset from current inference requests
  - Use the Evaluator component and the tools in TensorFlow Model Analysis
  - If necessary, retrain your model

Show

post_export_metrics/example_count

Sort by

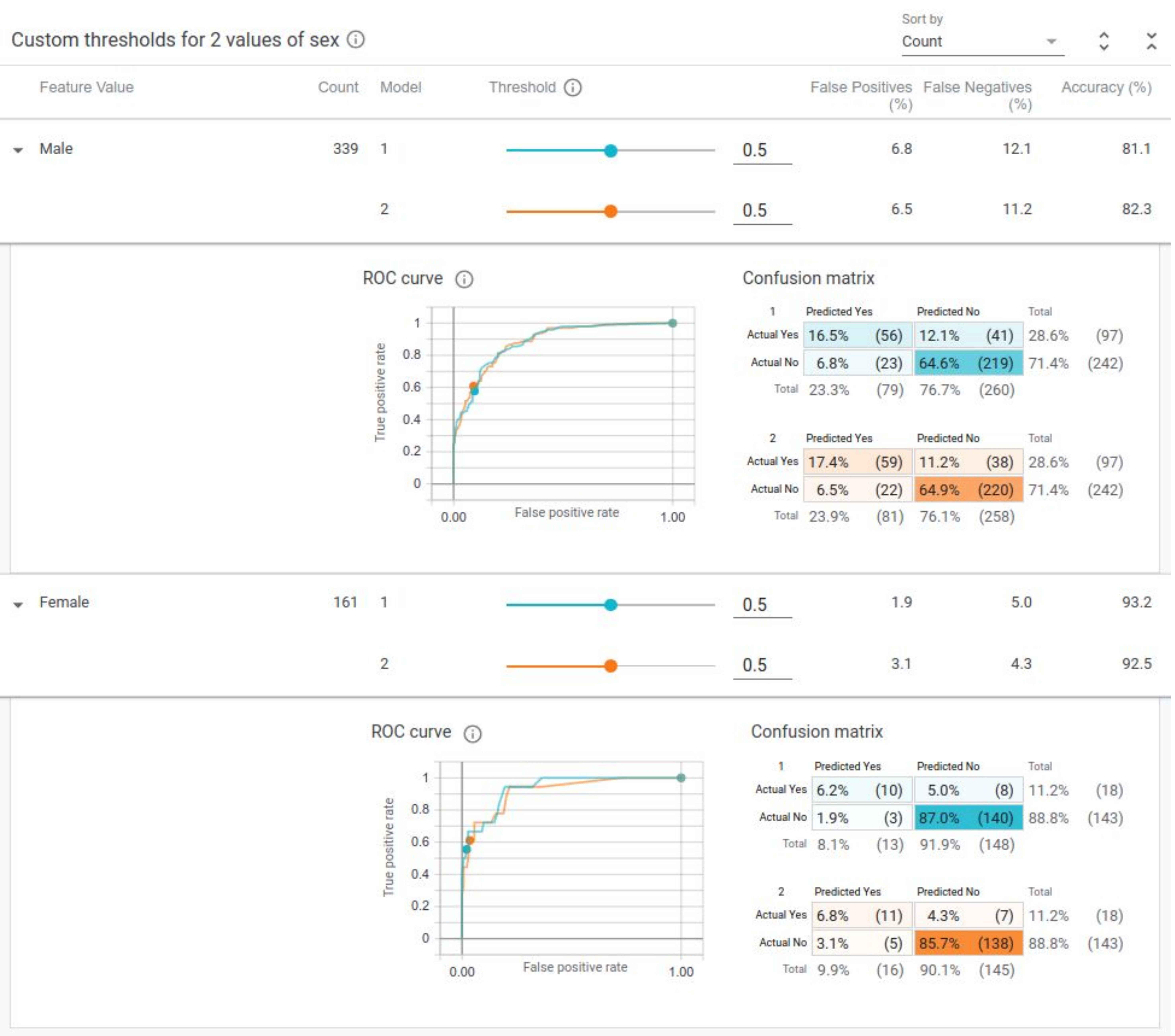
Slice



feature	accuracy	accuracy_baseline	auc	auc_precision_recall	average_lo
brick	0.74586	0.74033	0.95943	0.80808	0.358
light grey	0.79310	0.78325	0.95648	0.74818	0.324

Feature Space Coverage

- Identify regions in feature space where data coverage is sparse
- Collect more examples in sparse regions, if possible!
- Carefully add features to help create distinctions you'd like the model to make



Explore your model and data

What-if tool

Understand the input your model is receiving

Ask and answer “what-if” questions about your model’s output

Compare model performance across different slices of your data

Compare performance across multiple models

Quantify the Cost

Your model will never be 100%

- What does that extra performance cost?
- How does it affect different slices?

0.9573

TensorFlow Extended (TFX)

Standard components for your production model needs

Flexible orchestration and metadata

Extensible with custom components

TFX | TensorFlow

https://www.tensorflow.org/tfx

TensorFlow   Install   Learn   API   Resources   Community   Why TensorFlow   Search   GitHub

For Production

Overview   Tutorials   Guide   API

Apply to speak at TensorFlow World. Deadline April 23rd.   Propose talk

TensorFlow Extended (TFX) is an end-to-end platform for deploying production ML pipelines

When you're ready to move your models from research to production, use TFX to create and manage a production

https://www.tensorflow.org/tfx

Thank you!

Helpful resources

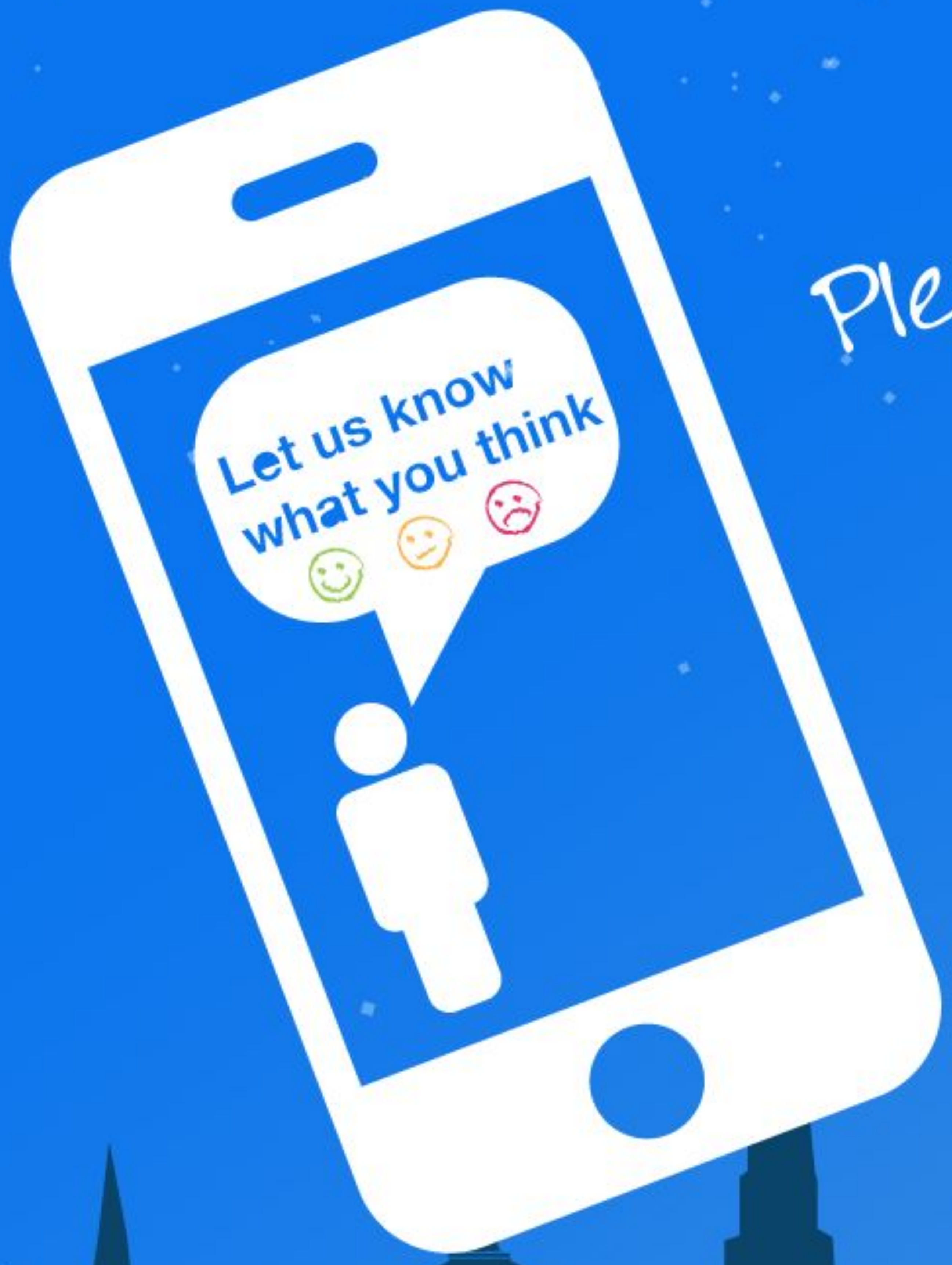
Web	https://tensorflow.org/tfx
Repo	https://github.com/tensorflow/tfx
Community	https://goo.gle/tfx-group
YouTube	https://goo.gle/tfx-youtube

Robert Crowe
TensorFlow Developer Advocate

@robert_crowe

goto; copenhagen

GOTO Copenhagen 2019
Conference Nov. 18 - 20



Please

Remember to rate this session

Thank you!

Follow us @gotocph

Images

https://freephotos.cc/shoes#404168

https://pixabay.com/photos/confused-hands-up-unsure-perplexed-2681507/

https://pixabay.com/photos/shoe-handmade-shoes-dress-shoes-632702/

https://pixabay.com/photos/shoes-brown-leather-fashion-2434210/

https://pixabay.com/photos/turtle-tortoise-reptile-2815539/

https://pixabay.com/photos/dog-and-cat-free-pet-cat-isolated-3484559/

https://pixabay.com/photos/sneakers-chuck-s-sneaker-shoe-2768263/

https://pixabay.com/photos/high-heeled-shoes-pumps-2781084/

https://pixabay.com/photos/business-stock-finance-market-1730089/

https://pixabay.com/vectors/tag-ticket-label-hole-color-35797/

https://pixabay.com/photos/isolated-hare-nature-animal-grass-2014108/