MATLAB EXPO 2018

Ampliando MATLAB Analytics con Kafka y Servicios en la Nube

Lucas García

Agenda

1 Access and Explore Data

- Files
- Databases
- Sensors

2 Preprocess Data

- Working with Messy Data
- Data Reduction/ Transformation
- Feature Extraction

3 Develop Predictive Models

- Model Creation e.g. Machine Learning
- Parameter Optimization
- Model Validation

4 Integrate with Production Systems

- Desktop Apps
- Enterprise Scale Systems
  - kafka
  - AWS Kinesis
  - OSIsoft.
- Embedded Devices and Hardware

5 Visualize Results

- 3rd party dashboards
  - Spotfire TIBCO Software
  - tableau
  - Qlik
  - Power BI
- Web apps

The Need for Large-Scale Streaming

Predictive Maintenance

Increase Operational Efficiency

Reduce Unplanned Downtime

Jet engine: ~800TB per day
Turbine: ~ 2 TB per day

More applications require near real-time analytics

Medical Devices

Patient Safety

Better Treatment Outcomes

Connected Cars

Safety, Maintenance

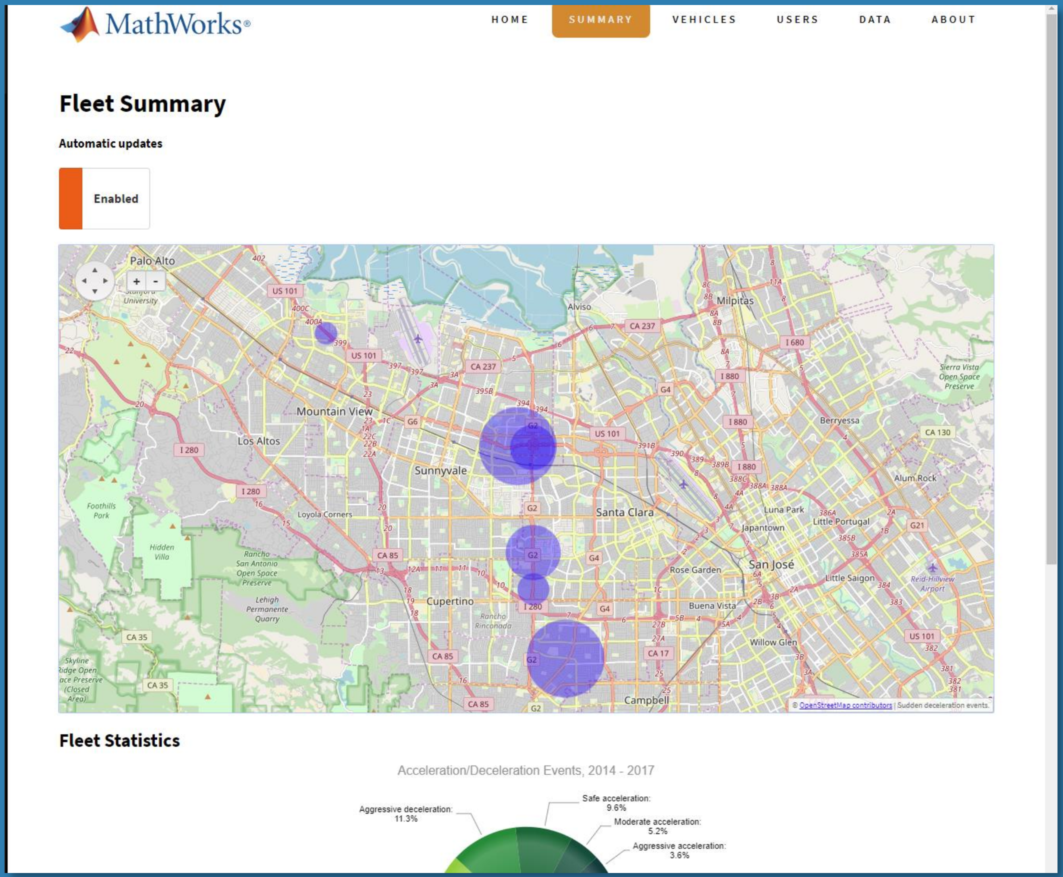
Advanced Driving Features

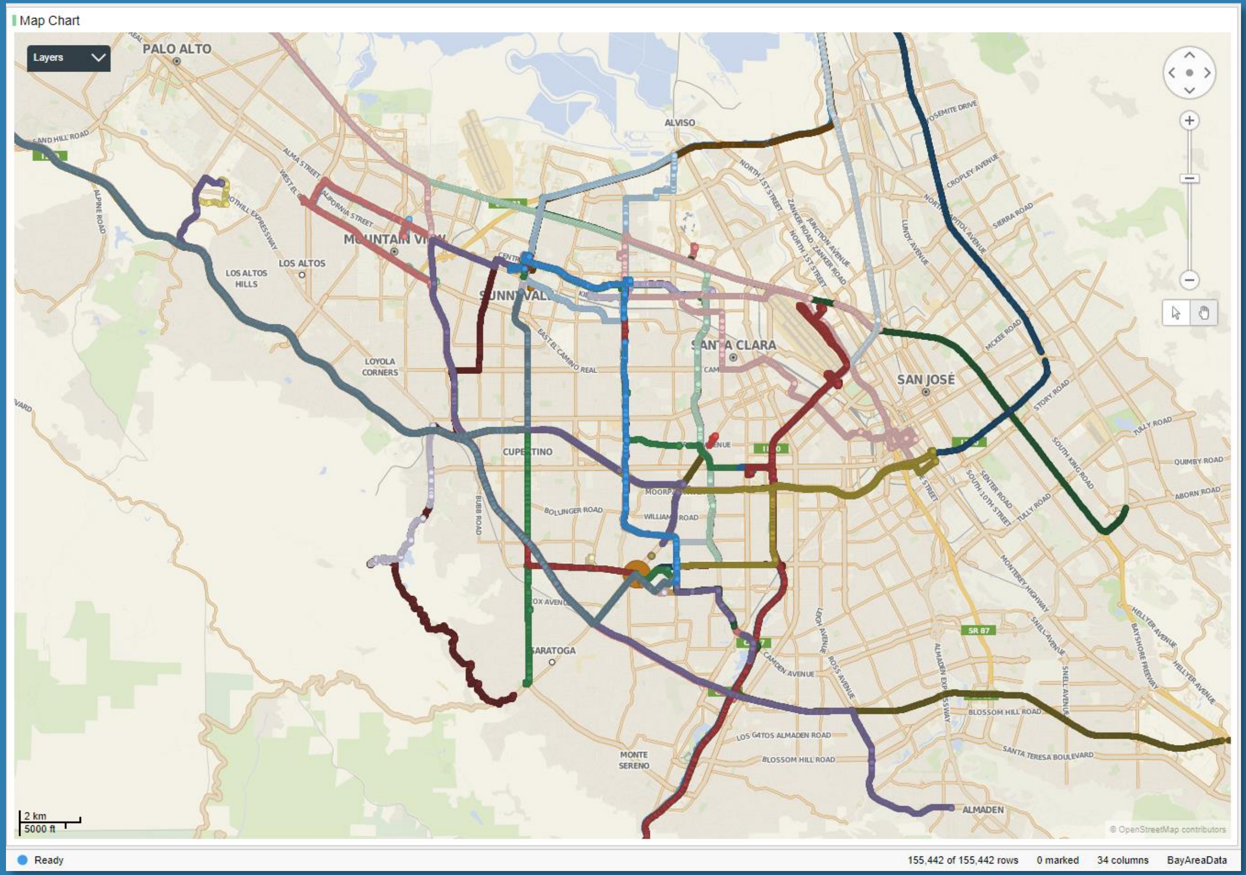
Car: ~25 GB per hour

Example Problem – How's my driving?

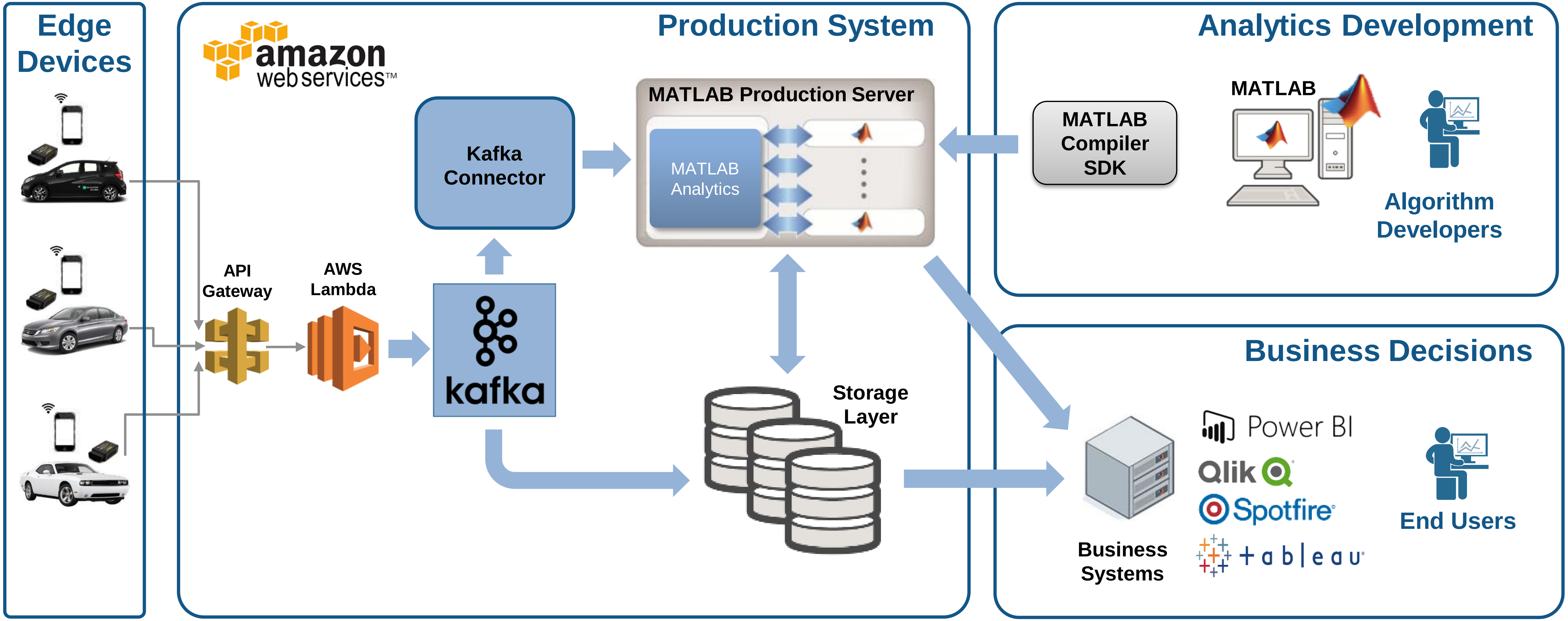
- A group of MathWorks employees installed an OBD dongle in their car that monitors the on-board systems
- Data is streamed to the cloud where it is aggregated and stored
- We would like to use this data to score the driving habits of participants

Example: Fleet Analytics with MATLAB



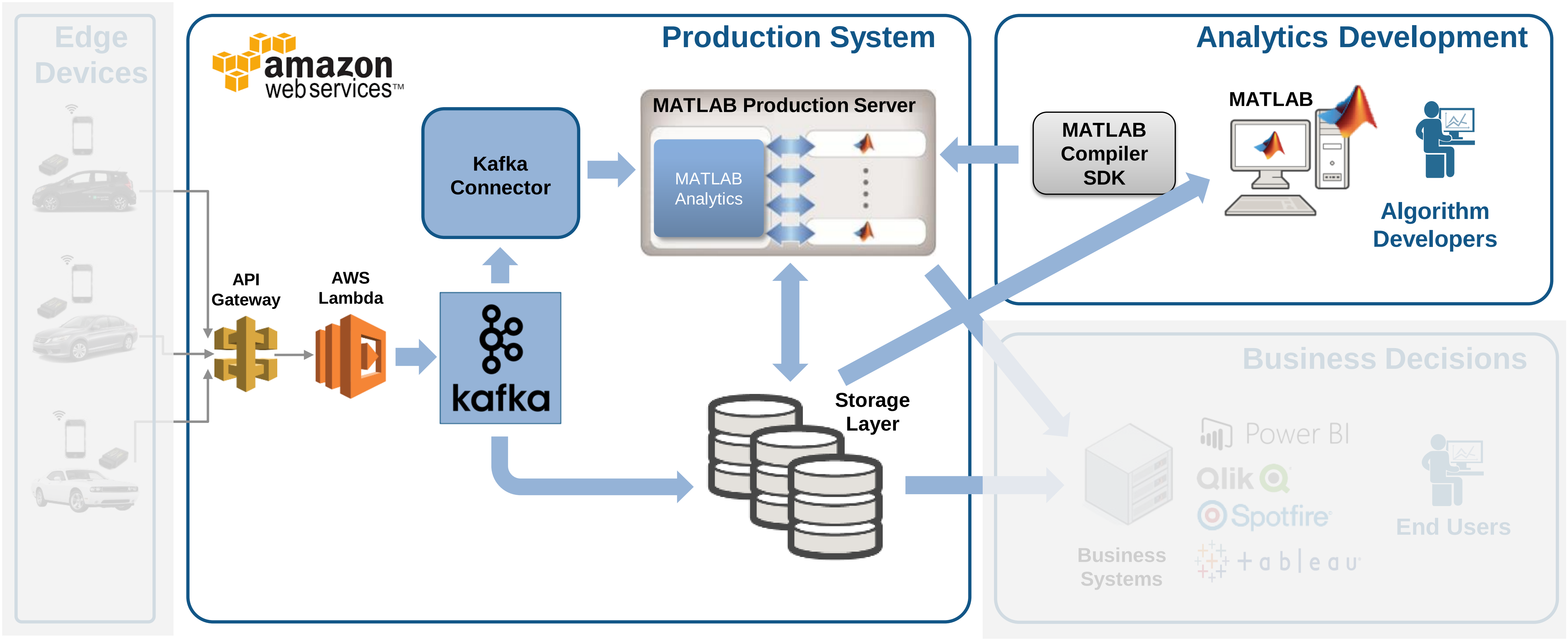


Fleet Analytics Architecture



1 Access and Explore Data

The first step is to clean up the incoming data



1

Access and Explore Data

The Data: Timestamped messages with JSON encoding


```
{
    "vehicles_id": {"$oid":"55a3fd0069702d5b41000000"},
    "time” : {"$date":"2015-07-13T18:01:35.000Z"},
    "kc” : 1975.0, "kff1225" : 100.65293, "kff125a" : 110.36619, …
}
```


Key

Timestamp

Values


```
{
    "vehicles_id": {"$oid":"55a3fe3569702d5c5c000020"}
    "time":{"$date":"2015-07-13T18:01:53.000Z"},
    "kc” : 2000.0, "kff1225" : 109.65293, "kff125a" : 115.36619,
     …
}
```



```
{
    "vehicles_id": {"$oid":"55a4193569702d115b000001"}
    "time":{"$date":"2015-07-12T19:04:04.000Z"}
    "kc":2200.0, "kff1225" : 112.65293, "kff125a" : 112.36619,
     …
}
```

Access a Sample of Data

✓ Decode JSON data
✓ Create Timetable

Raw Data

	timestamp	1 value	2 key
1	15-Jan-2015 22:12:23	'{ "_id" : { "$oid" : "55a41cb069702d115b059ee0" }, "trip_id" : { "$oid"...	'55a41cb069702d115b059ede'
2	15-Jan-2015 22:12:24	'{ "_id" : { "$oid" : "55a41cb069702d115b059ee1" }, "trip_id" : { "$oid"...	'55a41cb069702d115b059ede'
3	15-Jan-2015 22:12:25	'{ "_id" : { "$oid" : "55a41cb069702d115b059ee2" }, "trip_id" : { "$oid"...	'55a41cb069702d115b059ede'
4	15-Jan-2015 22:12:26	'{ "_id" : { "$oid" : "55a41cb069702d115b059ee3" }, "trip_id" : { "$oid"...	'55a41cb069702d115b059ede'

Timetable

t = 4647×40 *timetable*

	trip_id	VIN	kff1001	kff1005	kff1006	kff1220	kff1221	kff1222	kff1223	kff125a
1 Sun Jul 12 16:18:41 UTC 2015	55a3fe356...	55a3fe356...	17.1000	-84.9323	45.4704	NaN	NaN	NaN	NaN	59.0434
2 Sun Jul 12 16:18:42 UTC 2015	55a3fe356...	55a3fe356...	17.1000	-84.9322	45.4704	NaN	NaN	NaN	NaN	57.8609
3 Sun Jul 12 16:18:43 UTC 2015	55a3fe356...	55a3fe356...	18.9000	-84.9322	45.4705	NaN	NaN	NaN	NaN	52.7147
4 Sun Jul 12 16:18:44 UTC 2015	55a3fe356...	55a3fe356...	18.9000	-84.9322	45.4705	NaN	NaN	NaN	NaN	51.1983
5 Sun Jul 12 16:18:45 UTC 2015	55a3fe356...	55a3fe356...	18.0000	-84.9321	45.4706	NaN	NaN	NaN	NaN	49.1095
6 Sun Jul 12 16:19:13 UTC 2015	55a3fe356...	55a3fe356...	58.5000	-84.9305	45.4686	NaN	NaN	NaN	NaN	73.2005
7 Sun Jul 12 16:19:14 UTC 2015	55a3fe356...	55a3fe356...	56.7000	-84.9304	45.4685	NaN	NaN	NaN	NaN	75.3612
8 Sun Jul 12 16:19:15 UTC 2015	55a3fe356...	55a3fe356...	57.6000	-84.9304	45.4683	NaN	NaN	NaN	NaN	70.7542
9 Sun Jul 12 16:19:16 UTC 2015	55a3fe356...	55a3fe356...	56.7000	-84.9303	45.4682	NaN	NaN	NaN	NaN	62.8340

2

Preprocess Data

Develop a Preprocessing Function

Timetable

t = 4647×40 *timetable*

	trip_id	VIN	kff1001	kff1005	kff1006	kff1220	kff1221	kff1222	kff1223	kff125a
1 Sun Jul 12 16:18:41 UTC 2015	55a3fe356...	55a3fe356...	17.1000	-84.9323	45.4704	NaN	NaN	NaN	NaN	59.0434
2 Sun Jul 12 16:18:42 UTC 2015	55a3fe356...	55a3fe356...	17.1000	-84.9322	45.4704	NaN	NaN	NaN	NaN	57.8609
3 Sun Jul 12 16:18:43 UTC 2015	55a3fe356...	55a3fe356...	18.9000	-84.9322	45.4705	NaN	NaN	NaN	NaN	52.7147
4 Sun Jul 12 16:18:44 UTC 2015	55a3fe356...	55a3fe356...	18.9000	-84.9322	45.4705	NaN	NaN	NaN	NaN	51.1983
5 Sun Jul 12 16:18:45 UTC 2015	55a3fe356...	55a3fe356...	18.0000	-84.9321	45.4706	NaN	NaN	NaN	NaN	49.1095
6 Sun Jul 12 16:19:13 UTC 2015	55a3fe356...	55a3fe356...	58.5000	-84.9305	45.4686	NaN	NaN	NaN	NaN	73.2005
7 Sun Jul 12 16:19:14 UTC 2015	55a3fe356...	55a3fe356...	56.7000	-84.9304	45.468	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
8 Sun Jul 12 16:19:15 UTC 2015	55a3fe356...	55a3fe356...	57.6000	-84.9304	45.468	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]
9 Sun Jul 12 16:19:16 UTC 2015	55a3fe356...	55a3fe356...	56.7000	-84.9303	45.468	[illegible]	[illegible]	[illegible]	[illegible]	[illegible]

- ✓ Clean up
- ✓ Enrich
- ✓ Restructure

Preprocess data

```
t = sortrows(t);
t = rmmissing(t,'MinNumMissing',width(t)-2);
```

Perform windowed calculations

```
t.Speed = movmedian(t.SpeedGPS,3);
t.D1 = [0;diff(t.SpeedGPS)];
```

```
[tmin,tmax] = bounds(t.time);
tnew = tmin:seconds(10):tmax;
countsByTime = retime(t(:,'Event'),tnew,@histcounts);
```

1 Access and Explore Data

Ad Hoc Access to Data from MATLAB

athenaQuery.mlx

Access the data in S3

Bring up the AthenaClient

```
athenaClient = aws.athena.Client();
athenaClient.Database = 'trainingdata';
athenaClient.initialize();
```

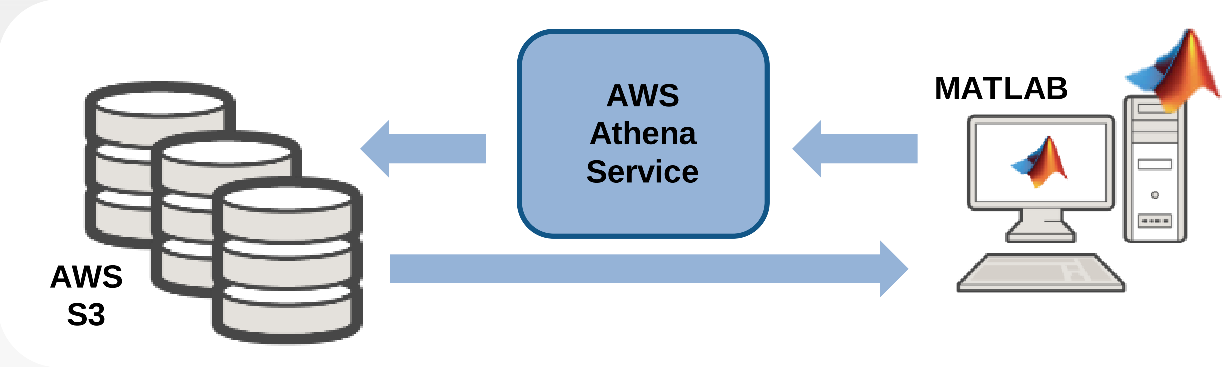
Create a query and submit

```
athenaClient.submitQuery('SELECT * FROM "trainingdata"."sampledata" limit 100','s3://fleettrainingdata')
```

Fetch data as a table for easy analysis

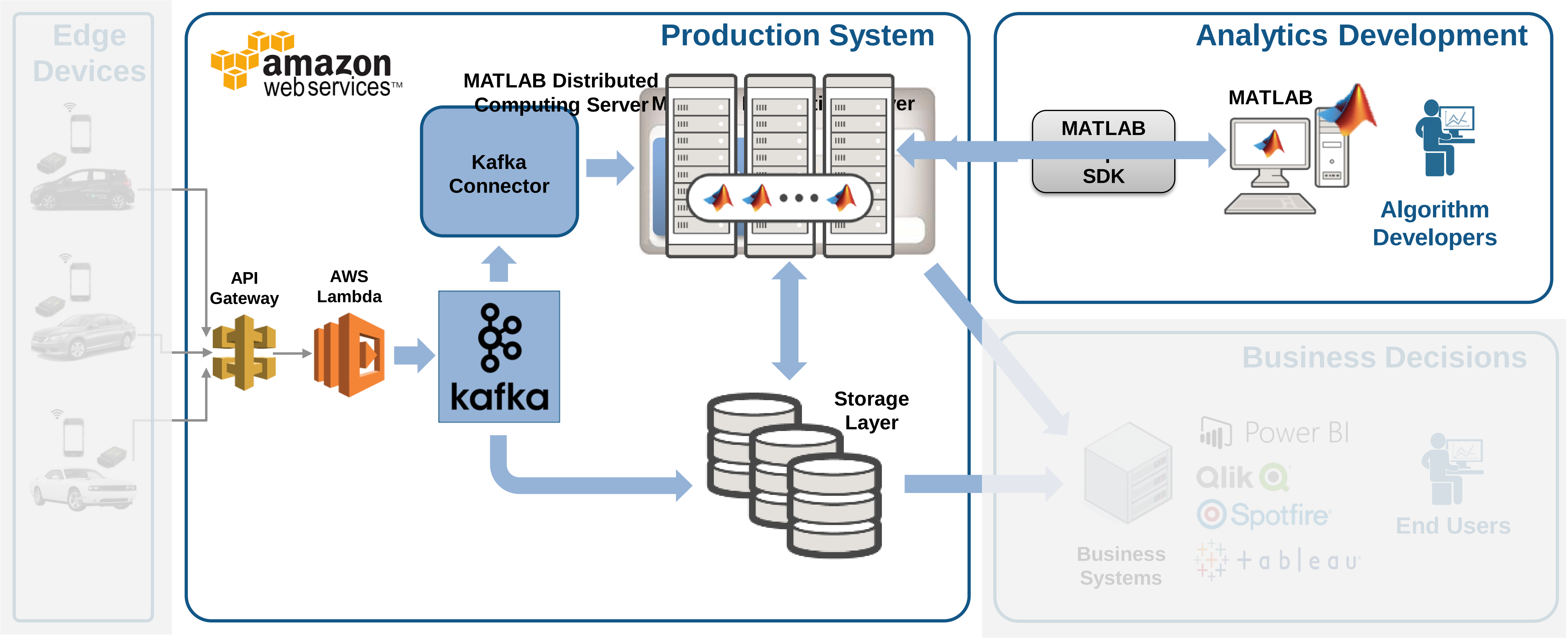
```
ds = datastore('s3://fleettrainingdata/*.csv');
ds.NumHeaderLines = 2;
data = table(ds);
```

Your usual MATLAB workflow goes here



3 Develop Predictive Models

Develop a Predictive Model



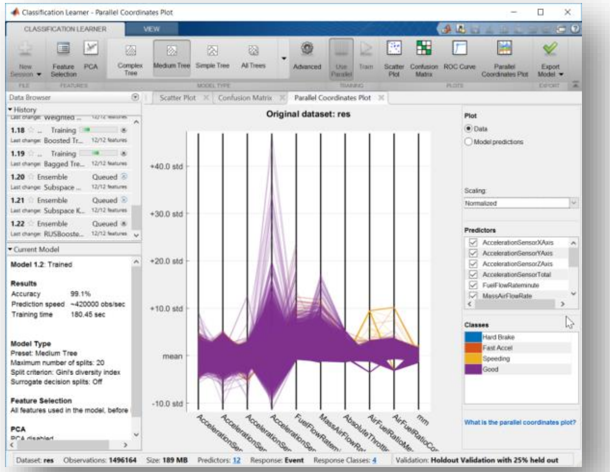
3

Develop Predictive Models

Everything you need to develop a predictive model is found in MATLAB

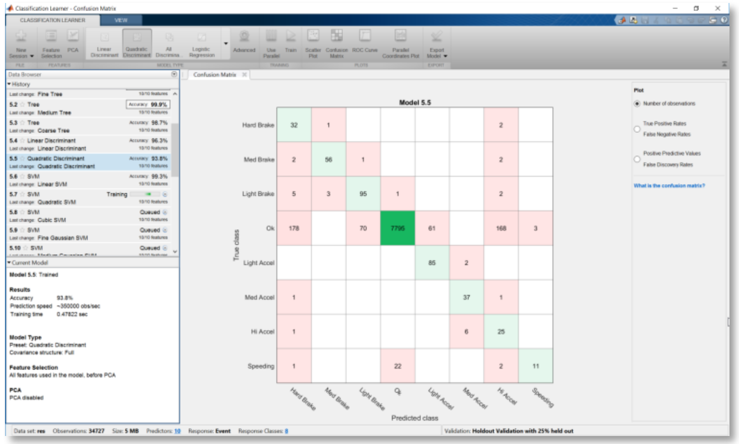
	1	2	3	4	5
time	Event	SpeedGPS	AccelerationSensorXAxis	AccelerationSensorYAxis	AccelerationSensorZAxi
Mon May 11 04:03:15 UTC 2015	Hard Brake	10.8360	-0.6996	0.6014	0.205
Wed May 06 19:09:48 UTC 2015	Hard Brake	27.8280	0.1419	0.9035	-0.526
Sun May 17 17:09:19 UTC 2015	Hard Brake	6.5520	0.9986	-0.0761	-0.004
Fri Jan 16 20:38:37 UTC 2015	Hard Brake	39.6128	0.0999	0.8000	0.367
Sat May 02 14:00:37 UTC 2015	Hard Brake	61.1280	0.4006	-0.4022	0.663
Mon Apr 27 17:54:27 UTC 2015	Fast Accel	37.7640	0.1527	0.4666	0.857
Sun May 03 21:00:42 UTC 2015	Fast Accel	17.2440	1.0235	0.0815	0.304
Mon May 04 11:30:33 UTC 2015	Fast Accel	19.6560	0.1336	0.8932	-0.578

Label Events



Represent Signals

Train Model



Validate Model

```
Evaluating tall expression using the Spark Cluster:
- Pass 1 of 2: Completed in 11 sec
- Pass 2 of 2: Completed in 2.3333 min
Evaluation completed in 2.6167 min
```

Scale up

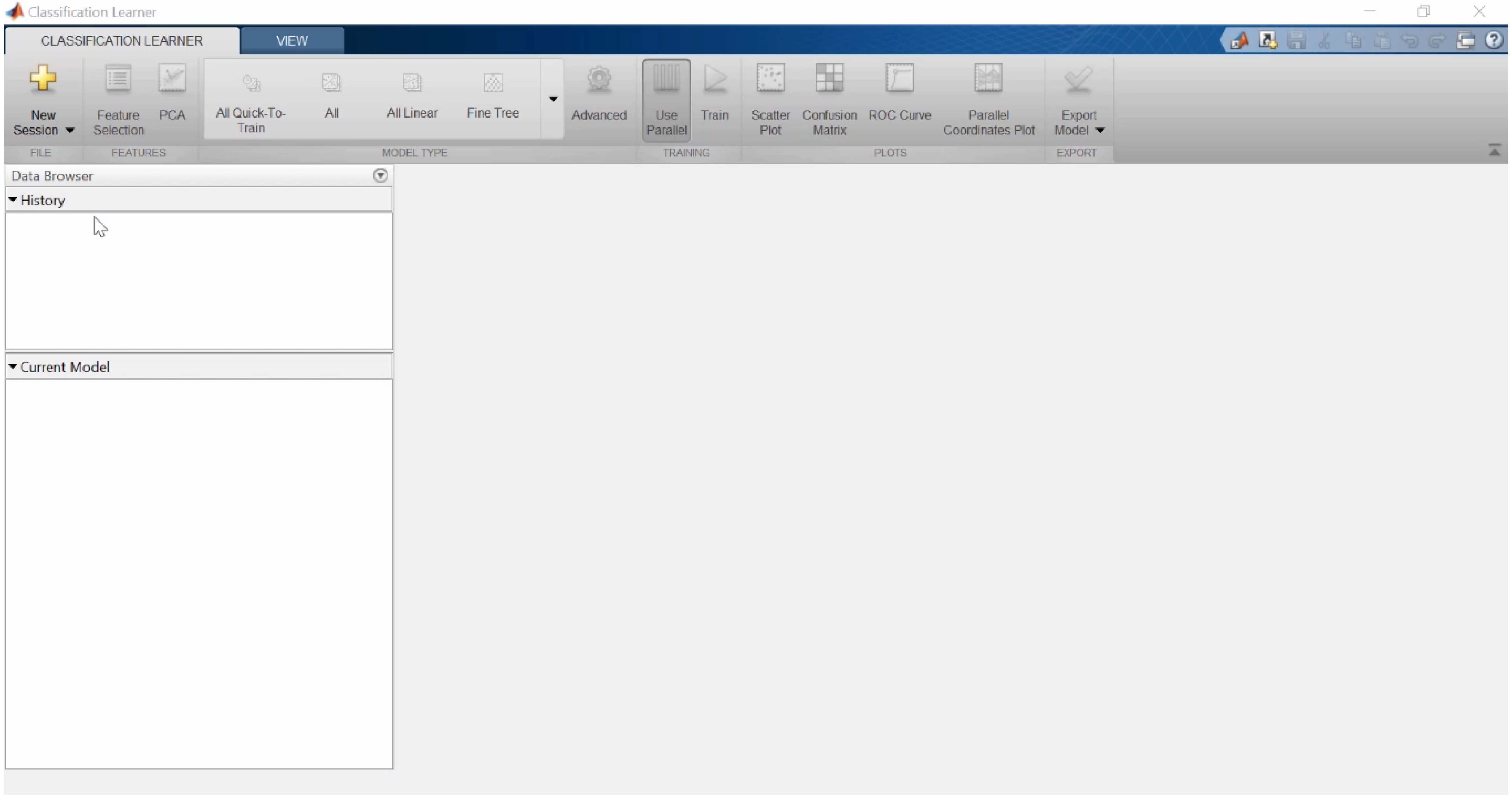
```
tt = tall(data);  % test tall array
model = TreeBagger(50,tt,'Event');
```

Scale to out of memory data

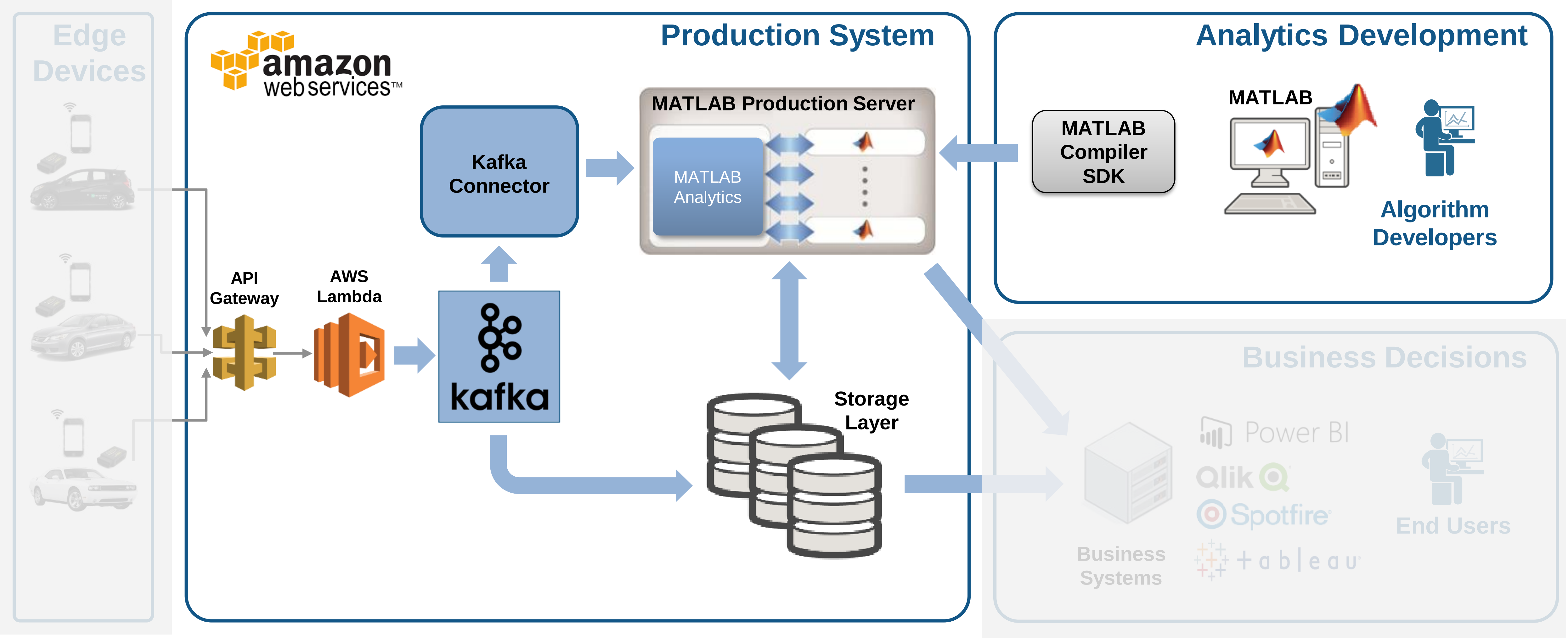
```
tt = tall(ds);
tt = preprocessData(tt);
model = TreeBagger(50,tt,'Event');
save machineLearningModel model
```

Scale Up

Develop a Predictive Model in MATLAB

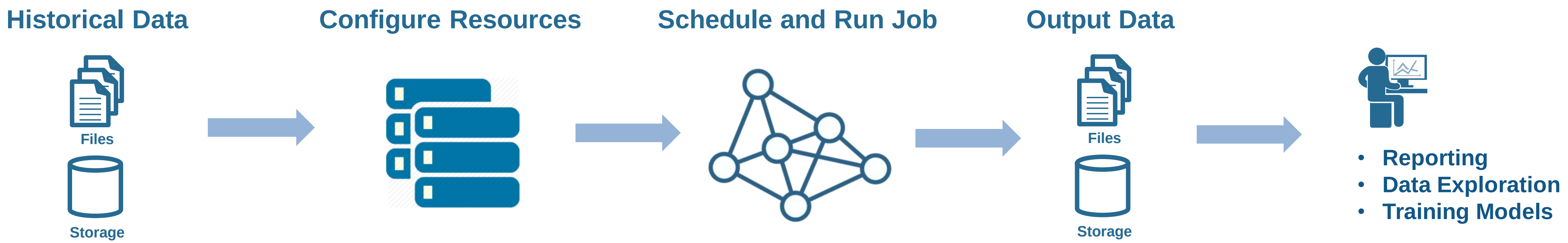


Integrate Analytics with Production Systems

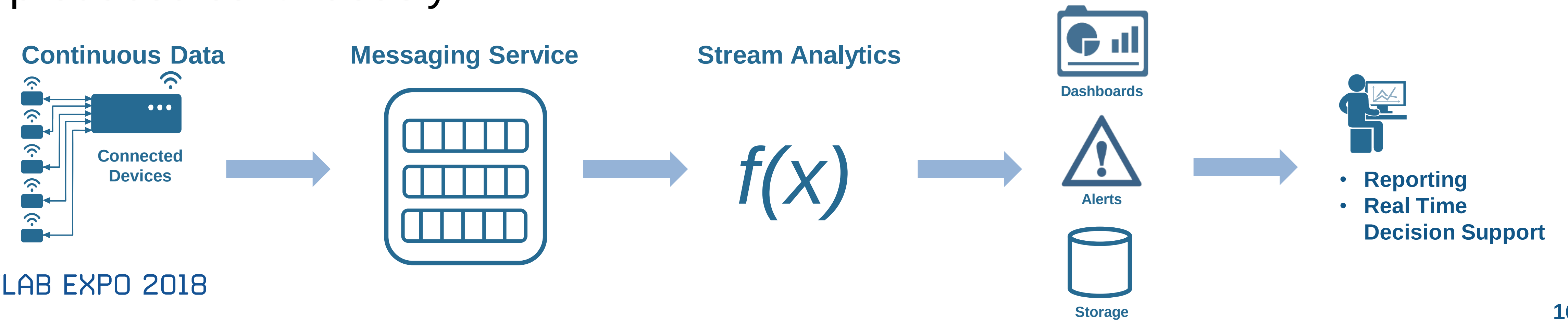


A quick Intro to Stream Processing

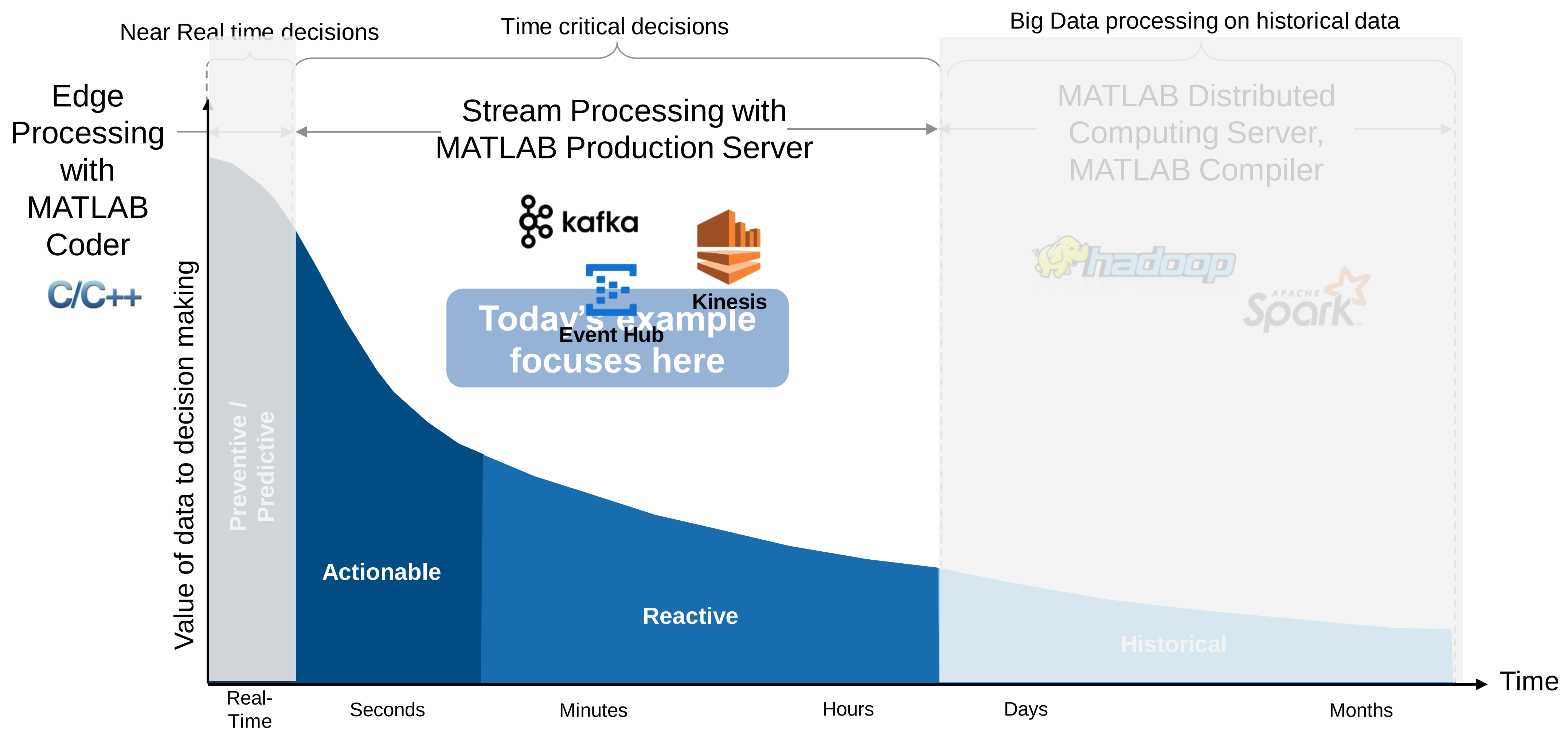
- ***Batch Processing*** applies computation to a finite sized historical data set that was acquired in the past



- ***Stream Processing*** applies computation to an unbounded data set that is produced continuously



Why stream processing?

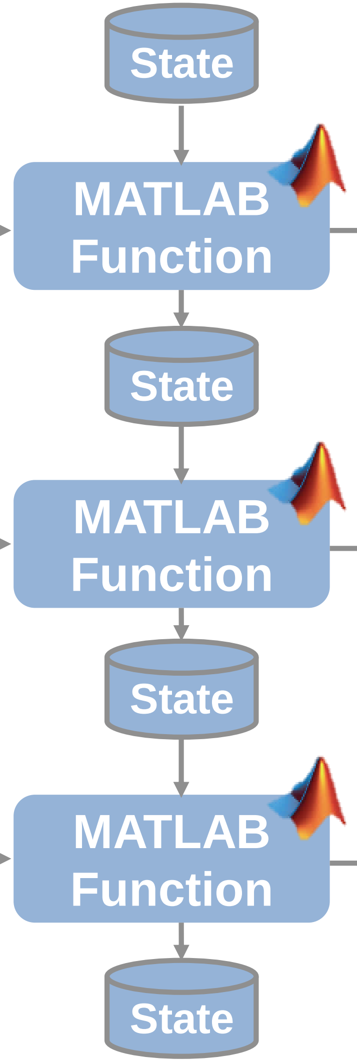


4 Integrate with Production Systems

Streaming data is treated as an unbounded Timetable

Input Table

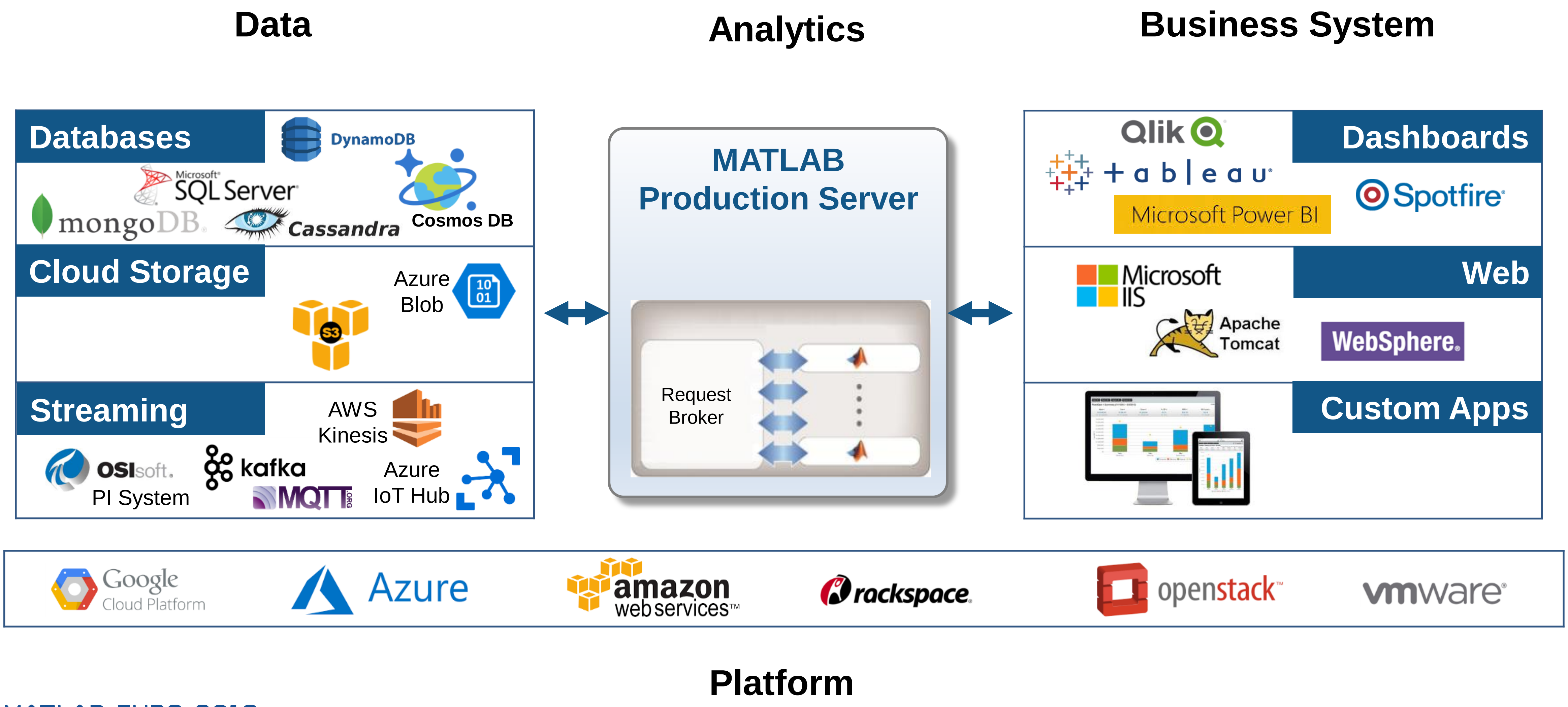
Event Time	Vehicle	RPM	Torque	Fuel Flow
18:01:10	55a3fd	1975	100	110
18:10:30	55a3fe	2000	109	115
18:05:20	55a3fd	1980	105	105
18:10:45	55a3fd	2100	110	100
18:30:10	55a419	2000	100	110
18:35:20	55a419	1960	103	105
18:20:40	55a3fe	1970	112	104
18:39:30	55a419	2100	105	110
18:30:00	55a3fe	1980	110	113
18:30:50	55a3fe	2000	100	110
…	…	…	…	…



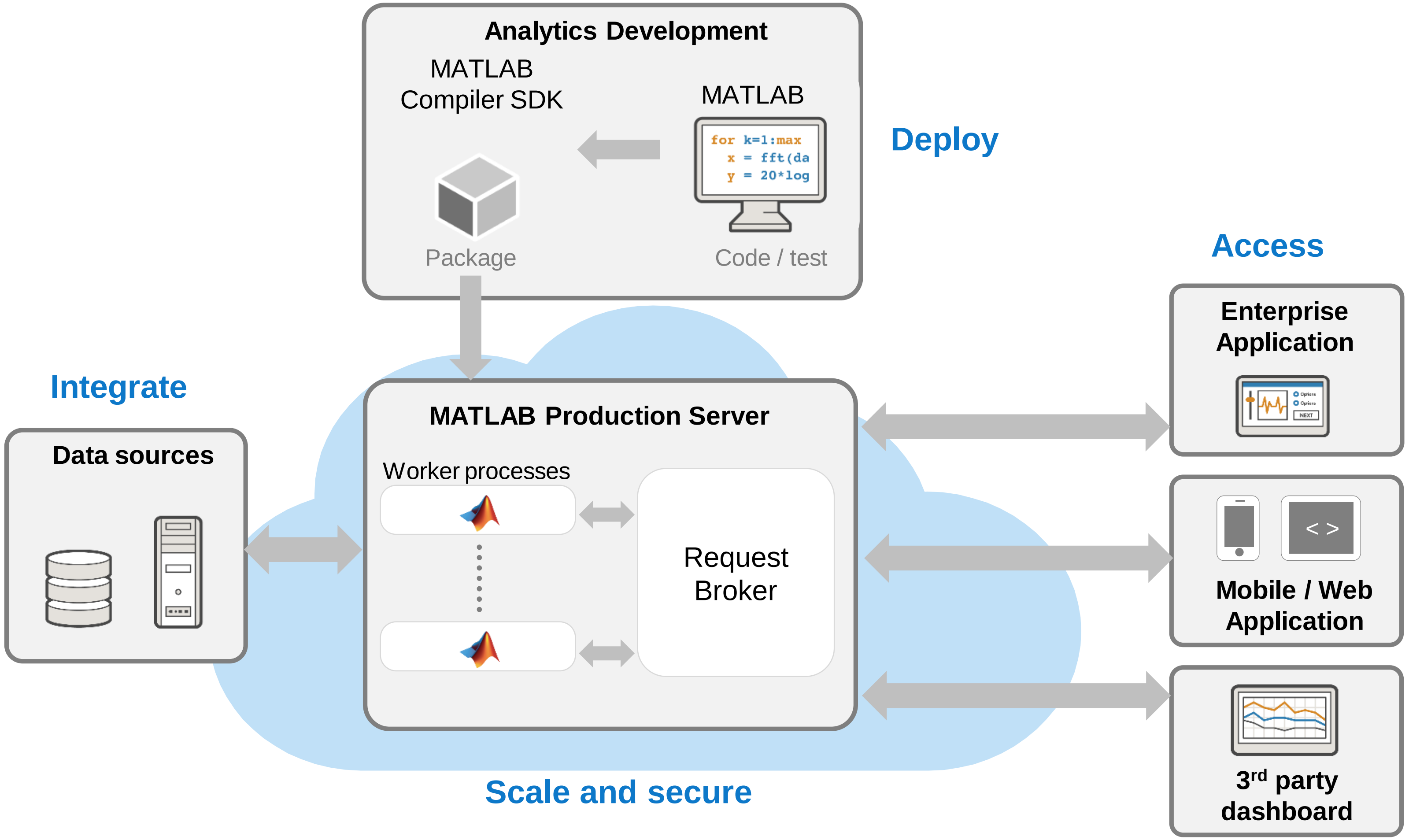
Output Table

Time window		Vehicle	Score
…		…	…
18:00:00	18:10:00	55a3fd	5
		55a3fe	…
		55a419	…
18:10:00	18:20:00	55a3fd	7
		55a3fe	3
		55a419	…
18:20:00	18:30:00	55a3fd	…
		55a3fe	4
		55a419	…
18:30:00	18:40:00	55a3fd	…
		55a3fe	5
		55a419	8

Introducing MATLAB Production Server

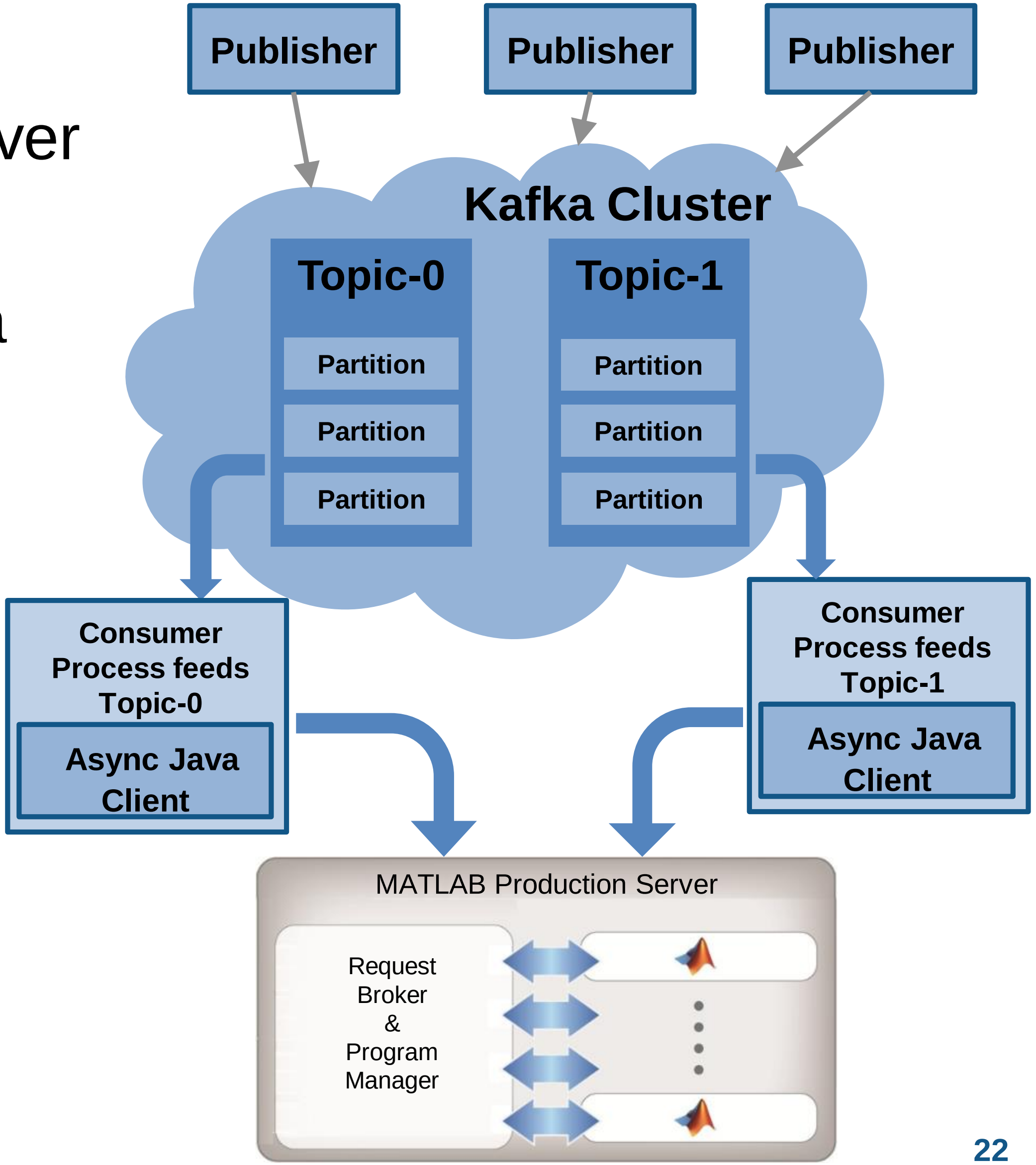


MATLAB Production Server is an application server that publishes MATLAB code as APIs

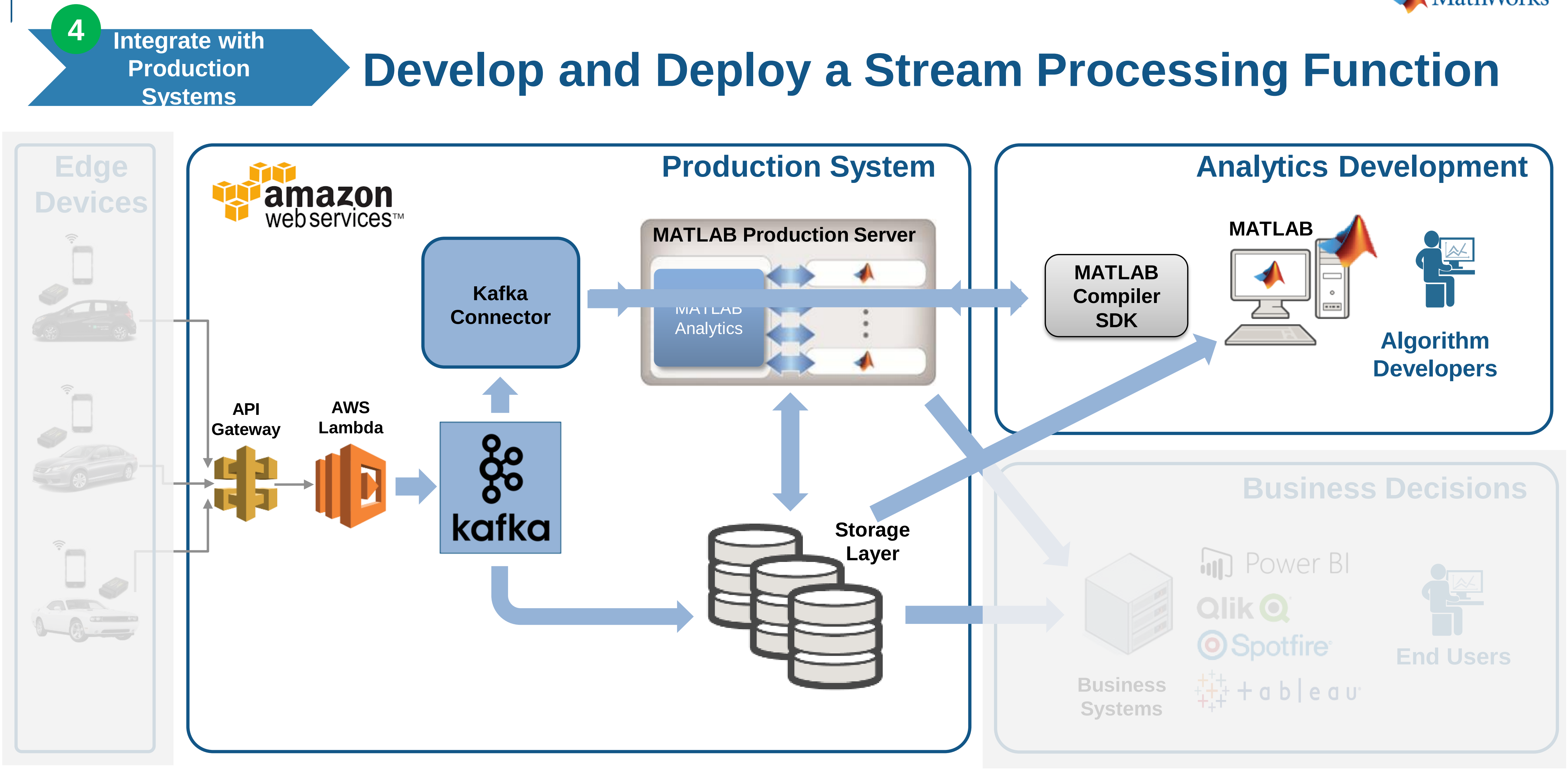


Connecting MATLAB Production Server to Kafka

- Kafka client for MATLAB Production Server feeds topics to functions deployed on the server
- Configurable batch of messages passed as a MATLAB Timetable
- Each consumer process feeds one topic to a specified function
- Drive everything from a simple config file
  - No programming outside of MATLAB!



Develop and Deploy a Stream Processing Function



Develop a Stream Processing Function in MATLAB

calculateScores.mlx

Develop a Streaming Function

```
function new_state = calculateScores(car_id, current_data, old_state, resultsStore)
```

Preprocess and perform calculations

```
current_data = preprocessData(current_data);
```

Predict driving events

```
current_data = predictEvents(current_data);
```

Count events for each ten second window

```
countsByTime = countEvents(current_data);
```

Write discrete data to mongodb

```
updateResultsStore(car_id,countsByTime,resultsStore);
```

Update new state

```
new_state = updateState(countsByTime,old_state);
end
```

Process each window of data as it arrives

Current score

Previous state

Current window of data to be processed

4 Integrate with Production Systems

Develop a Stream Processing Function in MATLAB

calculateScores.mlx

Develop a Streaming Function

```
function new_state = calculateScores(car_id, current_data
```

Preprocess and perform calculations

```
current_data = preprocessData(current_data);
```

Predict driving events

```
current_data = predictEvents(current_data);
```

Count events for each ten second window

```
countsByTime = countEvents(current_data);
```

Write discrete data to mongodb

```
updateResultsStore(car_id,countsByTime,resultsStore);
```

Update new state

```
new_state = updateState(countsByTime,old_state);
end
```

```
function current_data = preprocessData(current_data)
% Preprocess and perform calculations

% Remove records with all missing data
current_data = rmmissing(current_data,'MinNumMissing',width(current_data)-1);

% Smooth and calculate approximate gradients
current_data.Speed = movmedian(current_data.kff1001,5);
current_data.D1 = [0;diff(current_data.kff1001)];
current_data.D2 = [0;0;diff(current_data.kff1001,2)];
```

Apply your pre-processing algorithm

4 Integrate with Production Systems

Develop a Stream Processing Function in MATLAB

calculateScores.mlx

Develop a Streaming Function

```
function new_state = calculateScores(car_id, current_data, old_state, resultsStore)
```

Preprocess and perform calculations

```
current_data = preprocessData(current_data);
```

Predict driving events

```
current_data = predictEvents(current_data);
```

Count events for each ten second window

```
countsByTime = countEvents(current_data);
```

Write discrete data to mongodb

```
updateResultsStore(car_id,countsByTime,resultsStore);
```

Update new state

```
new_state = updateState(countsByTime,old_state);
end
```

```
function current_data = predictEvents(current_data)
% Predict events for current data based on machine learning model
predictorNames = {'kff1005','kff1006','kff125a','k10','kff1249','Speed','D1','D2',...
    'kff1001','kff1220','kff1221','kff1222','kff1223',...
    'k47','kff124d'};
predictors = current_data{:,predictorNames};
mdl = load('machineLearningModel.mat');
current_data.Event = predict(mdl.model,predictors);

end
```

Use the model you created with Classification Learner App

4 Integrate with Production Systems

Develop a Stream Processing Function in MATLAB

calculateScores.mlx

Develop a Streaming Function

```
function new_state = calculateScores(car_id, current_data, old_state, resultsStore)
```

Preprocess and perform calculations

```
current_data = preprocessData(current_data);
```

Predict driving events

```
current_data = predictEvents(current_data);
```

Count events for each ten second window

```
countsByTime = countEvents(current_data);
```

Write discrete data to mongodb

```
updateResultsStore(car_id,countsByTime,resultsStore);
```

Update new state

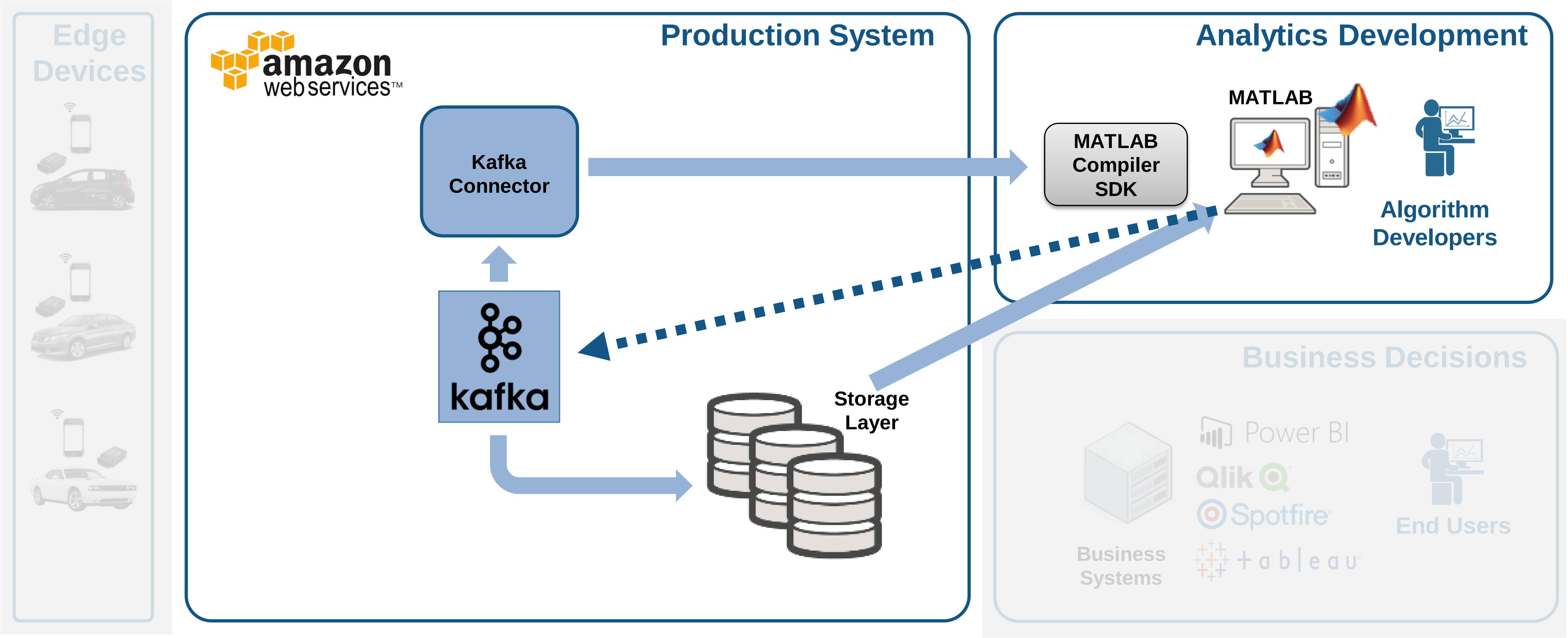
```
new_state = updateState(countsByTime,old_state);
end
```

Update Mongo database

- **Count of events by type and location**
- **Results of driver scoring**

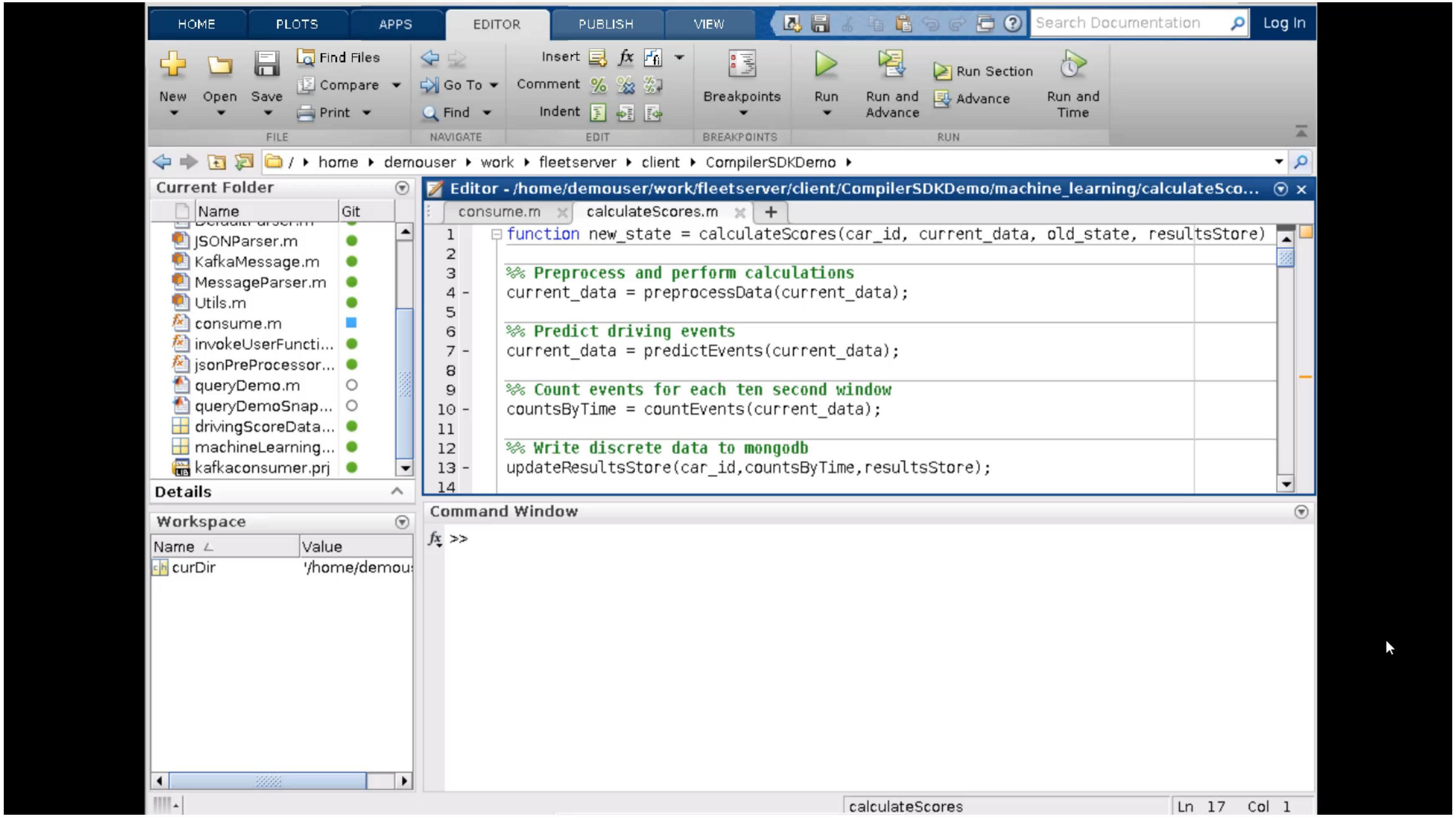
4 Integrate with Production Systems

Debug a Stream Processing Function in MATLAB



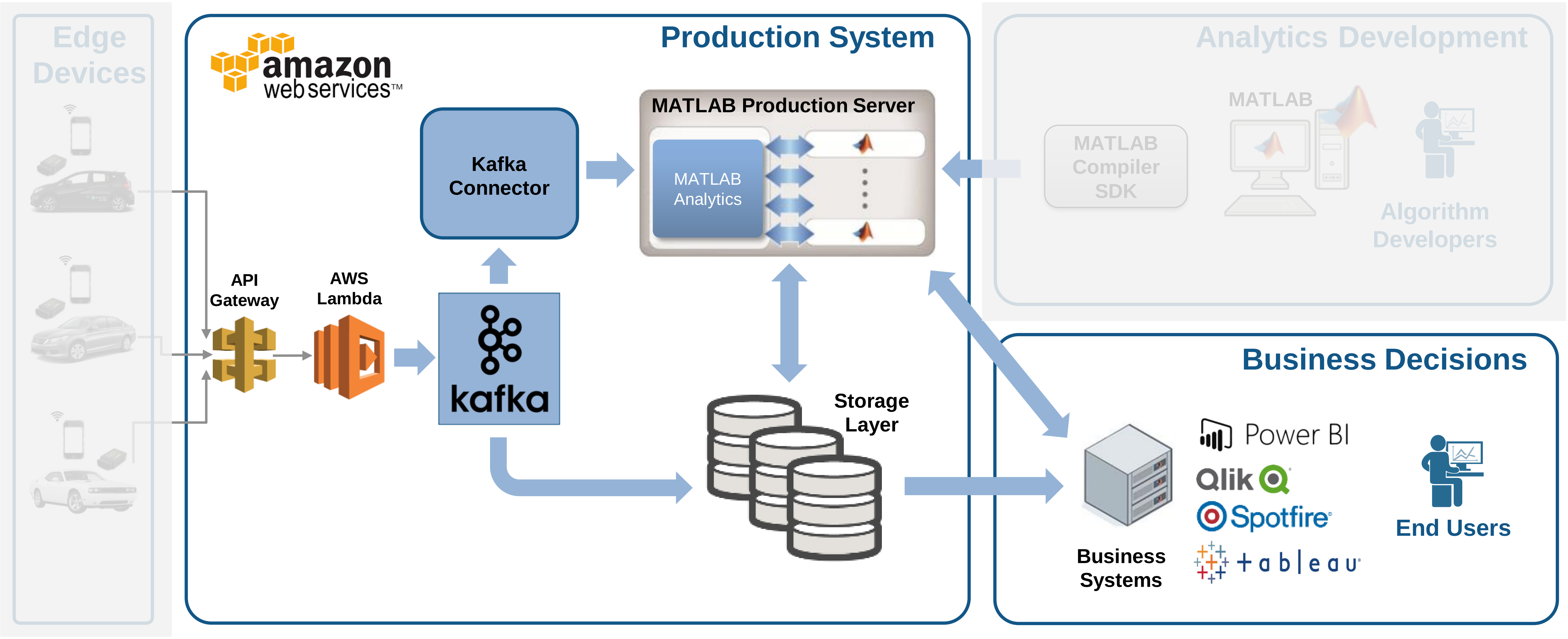
4 Integrate with Production Systems

Debug a Stream Processing Function in MATLAB



4 Integrate with Production Systems

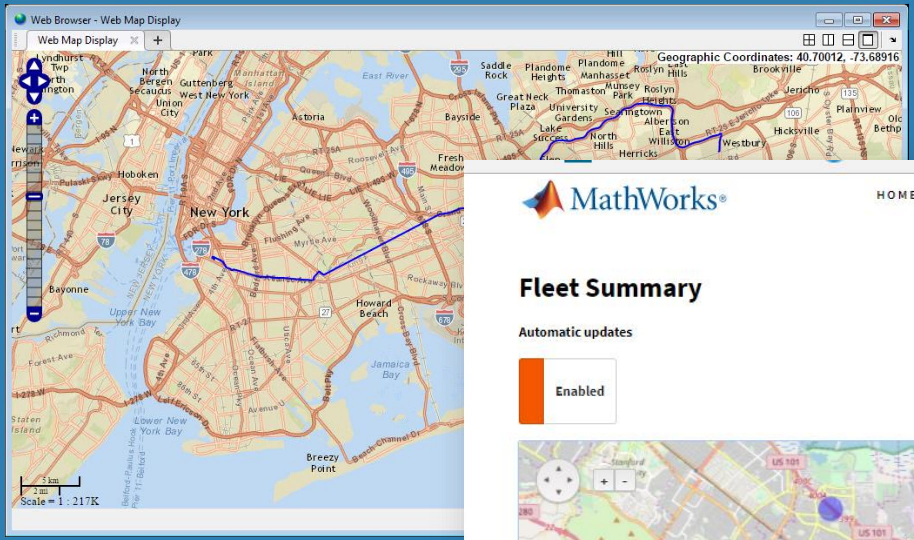
Tie in your Dashboard Application

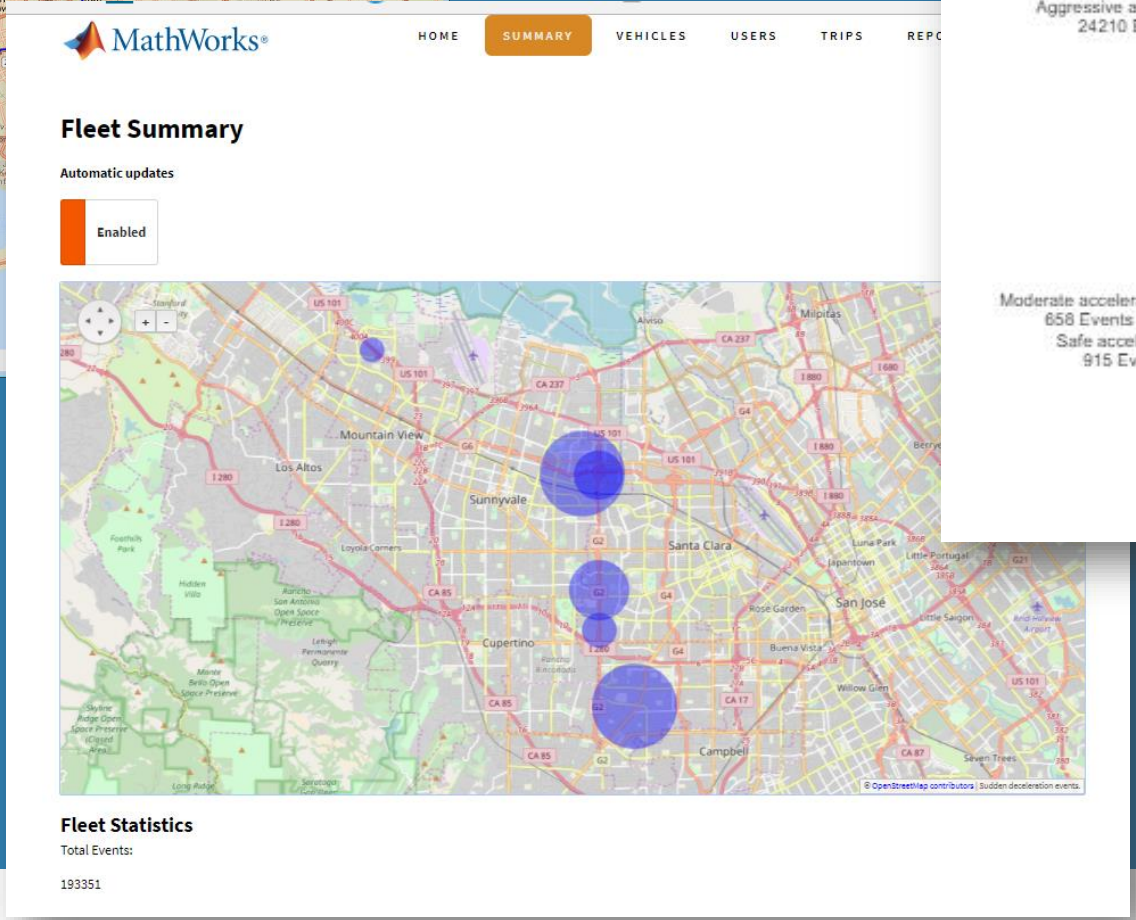


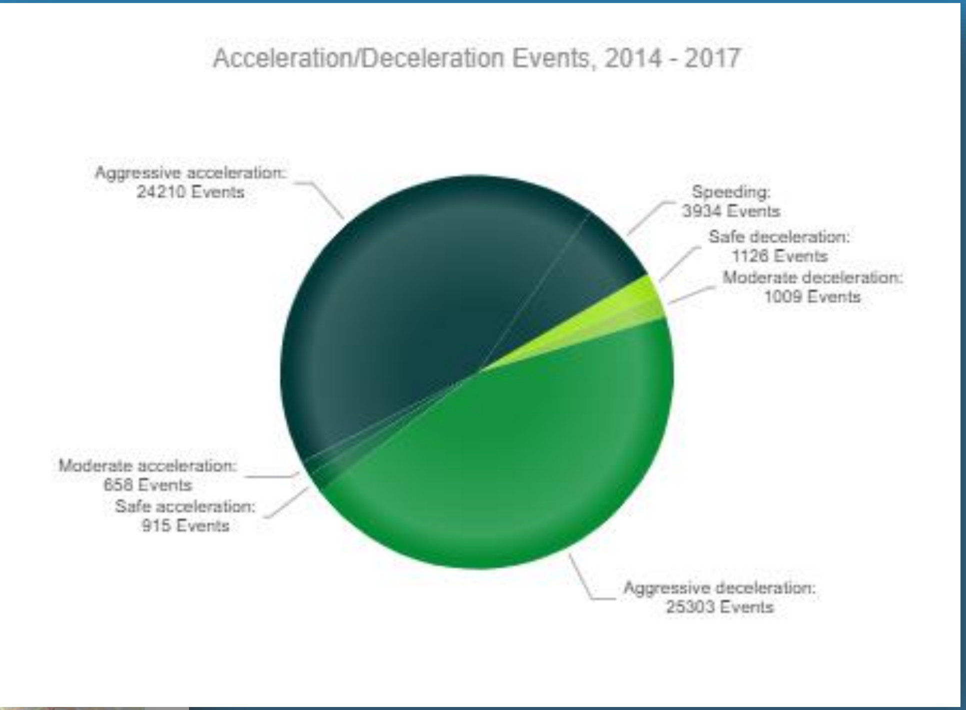
5

Visualize Results

Complete Your Application







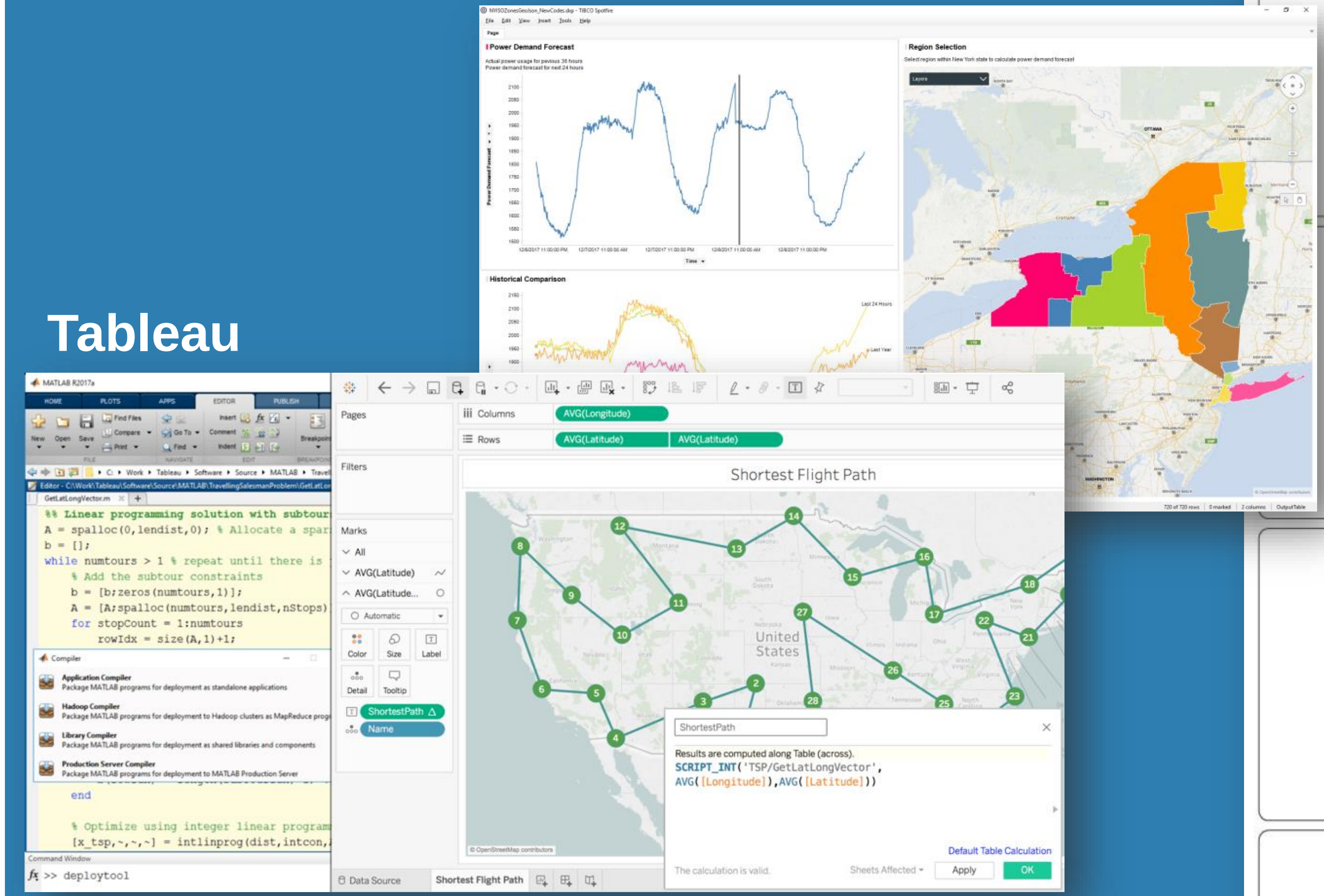
MathWorks

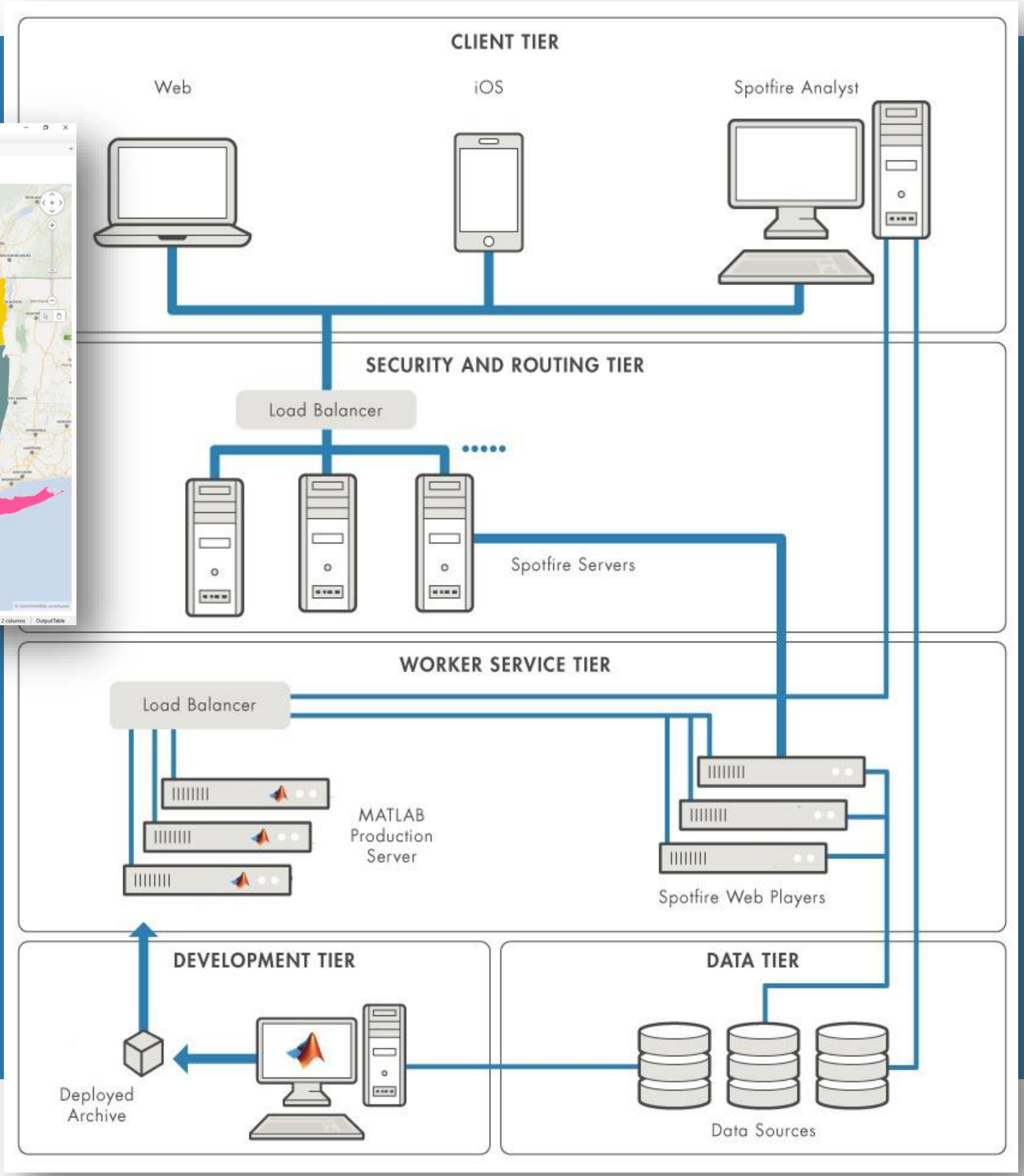
5 Visualize Results

Scalable Analytics with Enterprise BI Tools

TIBCO Spotfire

Tableau





Key Takeaways

- MATLAB connects directly to your data so you can quickly design and validate algorithms
- The MATLAB language and apps enable fast design iterations
- MATLAB Production Server enables easy integration of your MATLAB algorithms with enterprise production systems
- You to spend your time understanding the data and designing algorithms

Resources to learn and get started

- [Data Analytics with MATLAB]
- [MATLAB Production Server]
- [MATLAB Compiler SDK]
- [Statistics and Machine Learning Toolbox]
- [Database Toolbox]
- [Mapping Toolbox]
- [MATLAB with TIBCO Spotfire]
- [MATLAB with Tableau]
- [MATLAB with MongoDB]