Building a Unified Machine Learning Pipeline with XGBoost and Spark

Nan Zhu
Distributed Machine Learning Community (DMLC) & Microsoft

About Me

- Nan Zhu
  - Software Engineer in Microsoft
    - Spark Streaming, Structured Streaming integration with Azure Event Hubs ***(a talk at 5:00 p.m. today)***
    - Spark Workload Performance Test/Monitoring/Optimization
  - Committee member of Apache MxNet (incubator) and DMLC, Contributor of Apache Spark

About Distributed Machine Learning Community (DMLC)

- DMLC is a group of researchers and engineers collaborating on open-source machine learning projects
- What we are building
  - XGBoost (https://github.com/dmlc/xgboost)
  - MxNet (https://github.com/dmlc/mxnet)
  - Etc.

Agenda

- Introduction to XGBoost and XGBoost-Spark
  - Will not go into algorithm details and formula derivations(http://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf)
- Why Integrating XGBoost and Spark?
- Design of XGBoost-Spark
- What we can learn from XGBoost-Spark

Disclaimer: Personal Contribution to XGBoost Project

Introduction to XGBoost

XGBoost & XGBoost-Spark (1)

- XGBoost
  - A Gradient Boost Tree System
  - Created by Tianqi Chen (PhD student in UW) in 2014
  - Today: Python, R, Java, Scala, C++ bindings. Runs on single machine, Hadoop, Spark, Flink and GPU.
- XGBoost-Spark
  - Integrating XGBoost and Apache Spark
  - Idea generated during NIPS 2015 in the discussion between Tianqi and me.
  - First Generation (RDD) in March of 2016
  - Second Generation (DataFrame + ML Framework) in September of 2016

XGBoost & XGBoost-Spark (2)

- More than half of the winning solutions in machine learning challenges hosted at Kaggle adopt XGBoost
- XGBoost-Spark Users' Affiliations:
  - Airbnb, Alibaba, eBay, Microsoft, Snapshots, Tencent, Uber, etc.
- XGBoost Developers
  - University of Washington, Microsoft, Uptake, etc.

XGBoost: Gradient Boost Decision Tree

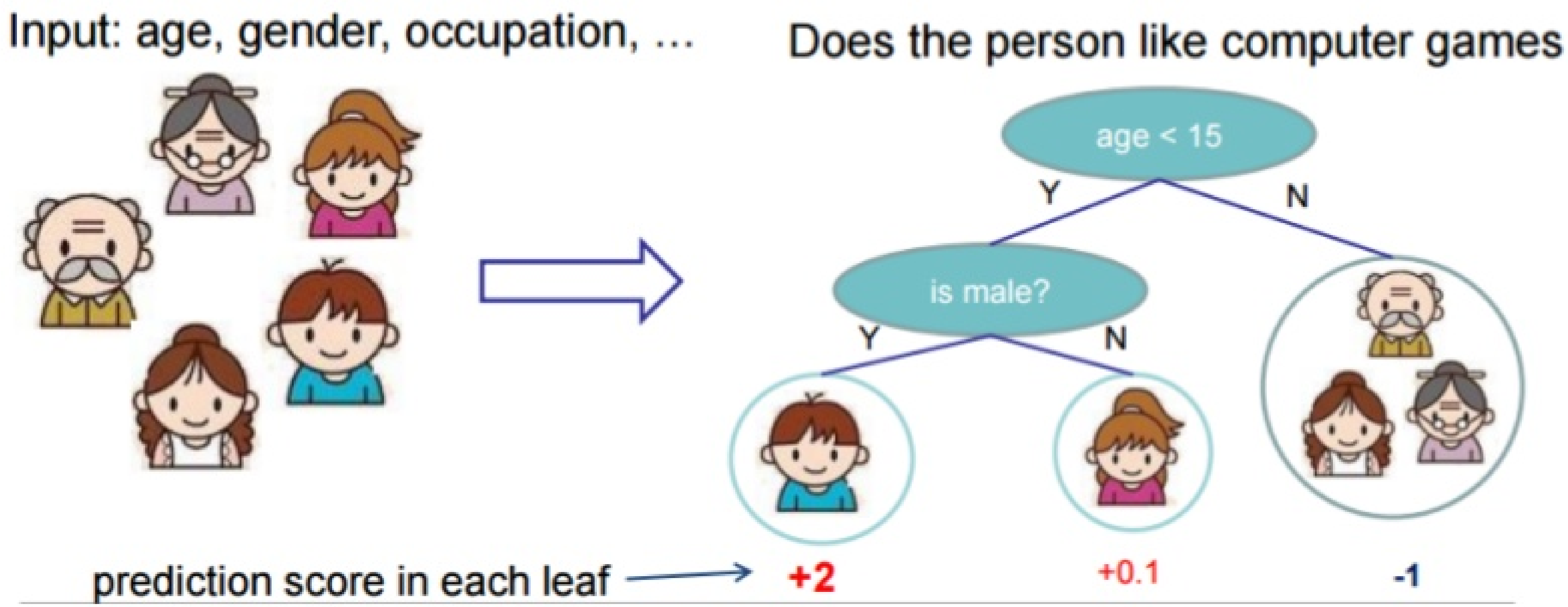
A

B

C

Decision Tree in XGBoost

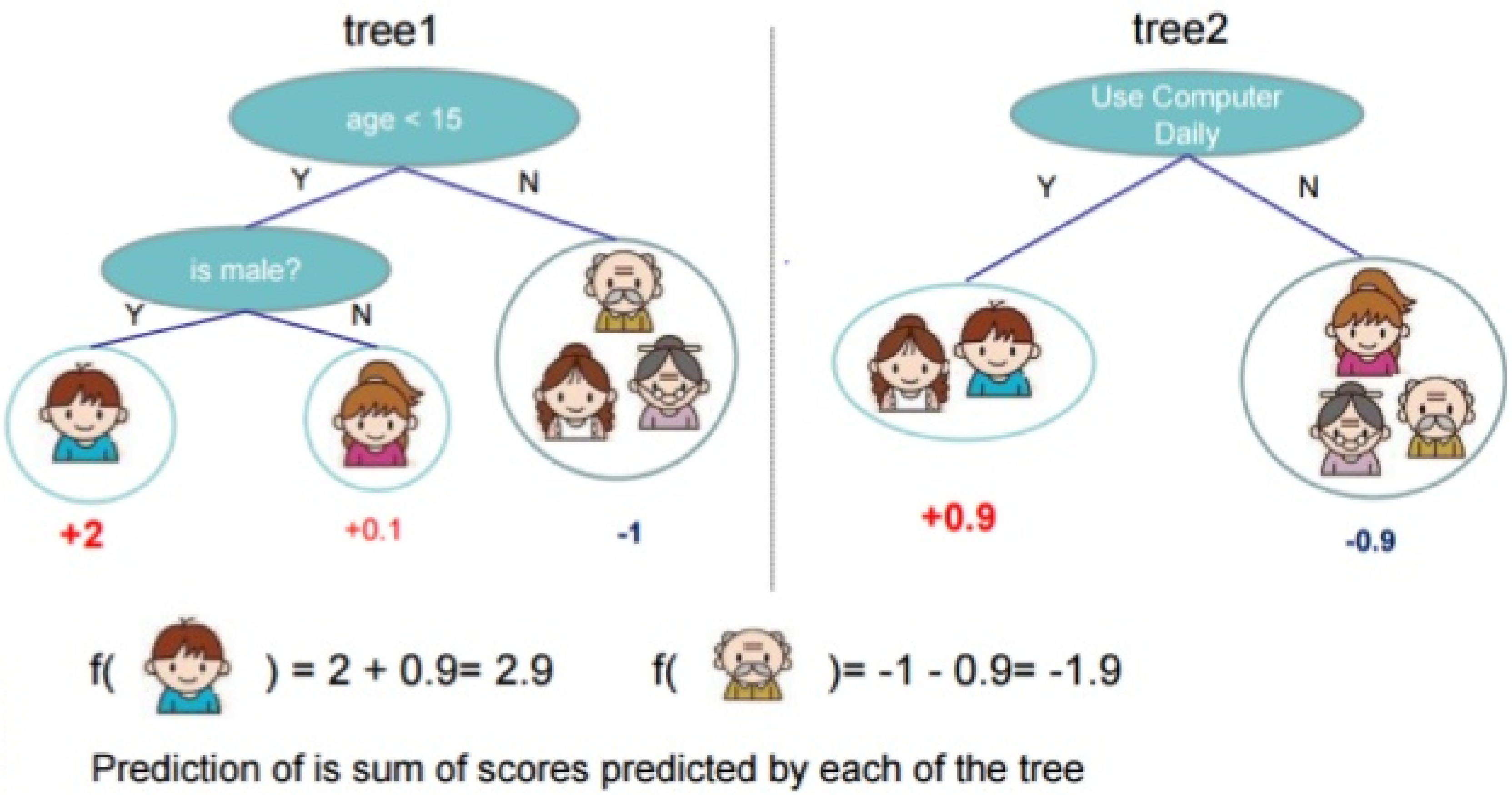
- CART: Classification and Regression Tree



NOTE: A lot of games are not tailored to girls' interests which need more improvements !!!

What is Decision Tree Boosting?

- Tree Boosting with CARTs

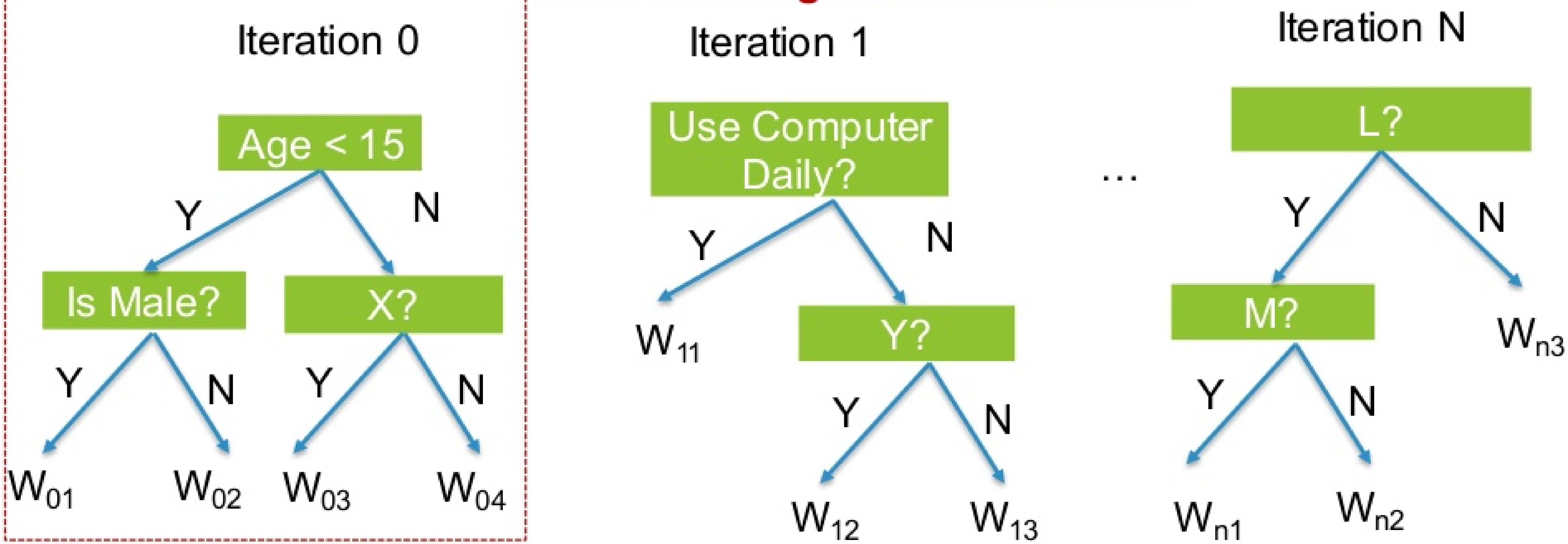


Ensemble Learning:

Use multiple weaker Learners to achieve better performance than anyone alone

Learning Trees with XGBoost

How: what is gradient boost tree?



Supervised Learning Basics

$$Obj(\theta) = L(\theta) + \Omega(\theta)$$

$L(\theta)$ - Training Loss: measures how well model fit on training data

$\Omega(\theta)$ - Regularization: measures complexity of model (we do not want to get a model only fitting with already-seen, i.e. training, data)

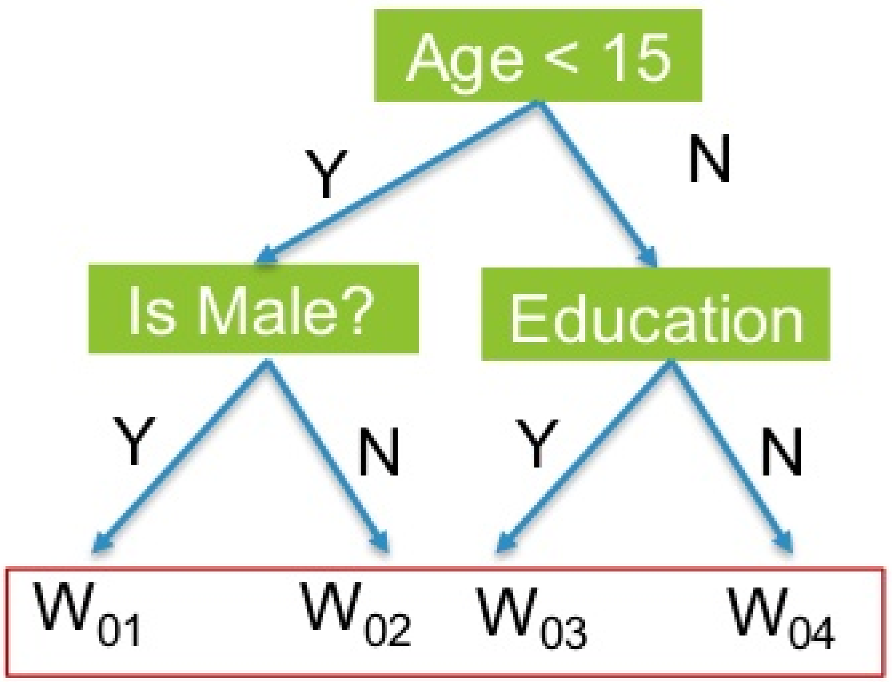
Objective Function in XGBoost

$$Obj^t(\theta) = \sum_{i=1}^{n} L(y_i, y_i^{t-1} + \boxed{f_t(x_i)}) + \Omega(f_t)$$

y_i - ground truth of data point i $\quad$ y_i^{t-1}- prediction for data point i in iteration t -1

f_t – tree with the optimal structure to be added in iteration t

Gradient Boosting in XGBoost (1)



Question 1: How to decide values of W_{0x}?

Gradient Boosting in XGBoost (2)

$$Obj^t(\theta) = \sum_{i=1}^{n} L(y_i, y_i^{t-1} + f_t(x_i)) + \Omega(f_t)$$

$f_t(x_i)$: ***a vector*** of scores of leave nodes, and ***a function*** maps data points to leaves, $w_q(x)$

$\Omega(f_t)$: number of leaf nodes, T, and sum of squared score of leaf nodes

$$= \sum_{i=1}^{n} L(y_i, y_i^{t-1} + w_q(x)) + \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

Gradient Boosting in XGBoost (3)

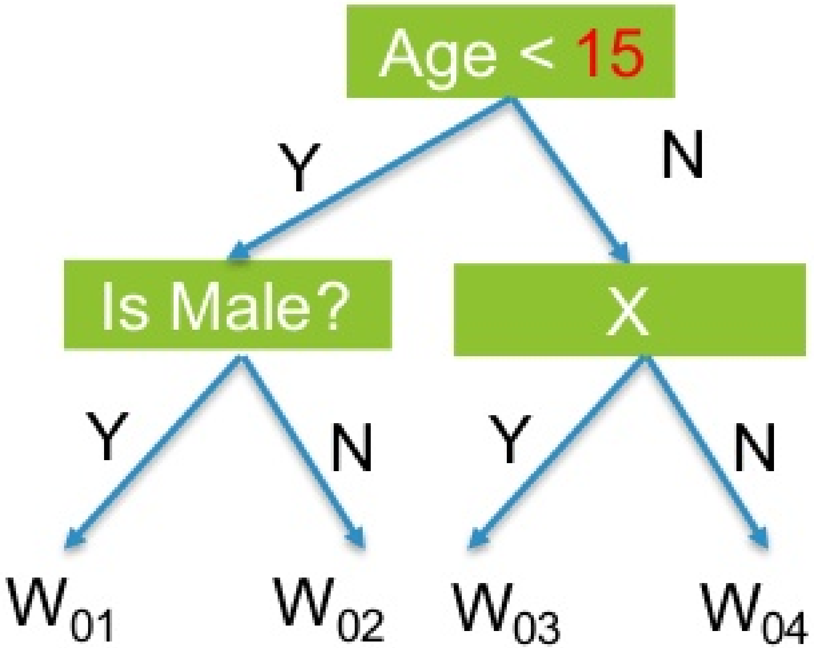
Finally $w_j^* = -\frac{G_j}{H_j+\lambda}$ $Obj = -\frac{1}{2}\sum_{j=1}^{T}\frac{G_j^2}{H_j+\lambda} + \gamma T$

where $G_j = \sum_{i\in I_j} g_i$ $H_j = \sum_{i\in I_j} h_i$

$$g_i = \partial_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)}), \quad h_i = \partial^2_{\hat{y}^{(t-1)}} l(y_i, \hat{y}^{(t-1)})$$

(http://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf)

Finding Optimal Splitting Point in XGBoost



Question 2: How to decide splitting point, e.g. 15?

(http://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf)

Approximate Algorithm to Find Best Splits

Avoid Enumerating every feature value in a node, as they might be continuous

For each feature k:

(1) Find candidate splitting points (S_{k1}, …, S_{kl}), (transforming continuous feature values to discrete buckets pursuing even distribution)

(2) Split with the maximum loss reduction corresponding splitting points

$$score \leftarrow \max(score, \frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{G^2}{H+\lambda})$$

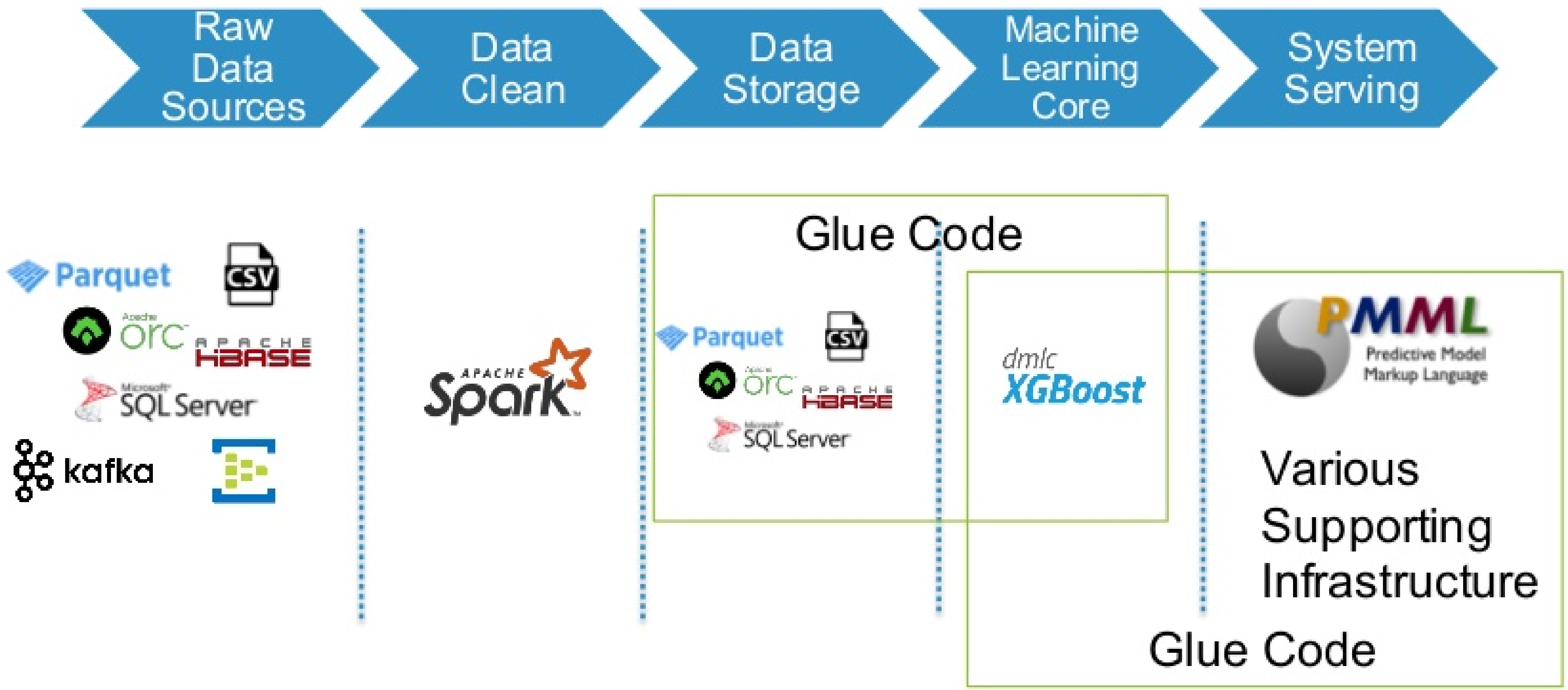
(http://www.kdd.org/kdd2016/papers/files/rfp0697-chenAemb.pdf)

Other optimizations in XGBoost

- Parallel Tree Construction
- Sparsity-aware Split Finding
  - Learning default direction for missing values from data
- Cache-aware Access
  - Memory prefetching
  - Cache-friendly thread working memory size
- Out-of-core computation
  - Scale to data size larger than physical memory
- Distributed Training

XGBoost is so good! Let's build a machine learning ***Pipeline*** based on XGBoost!!!

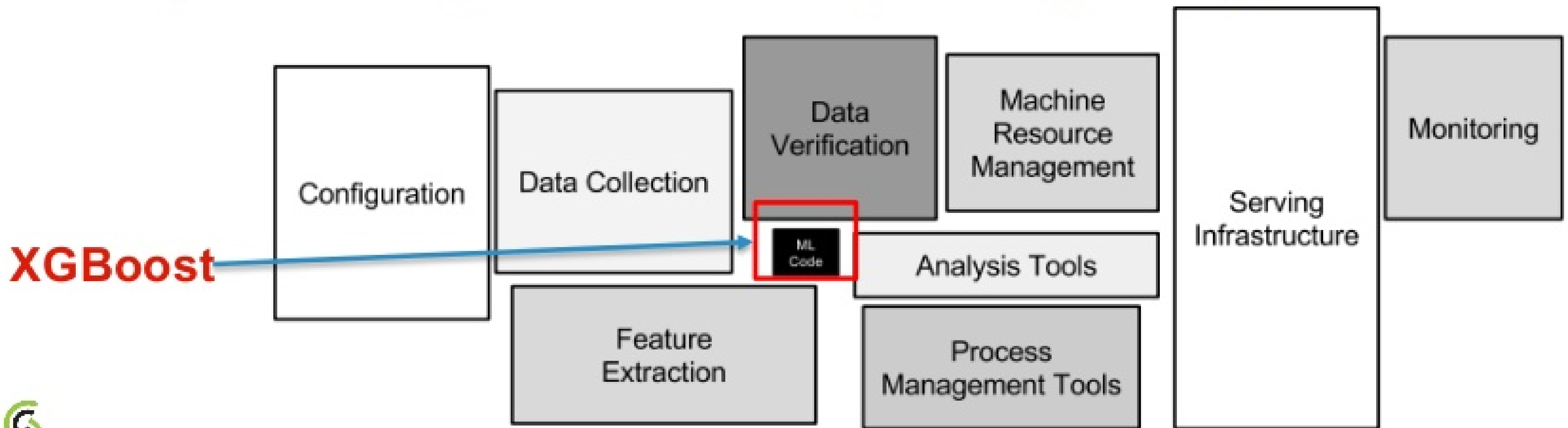
First Version (Separate XGBoost)



Painpoint in Productionalizing XGBoost

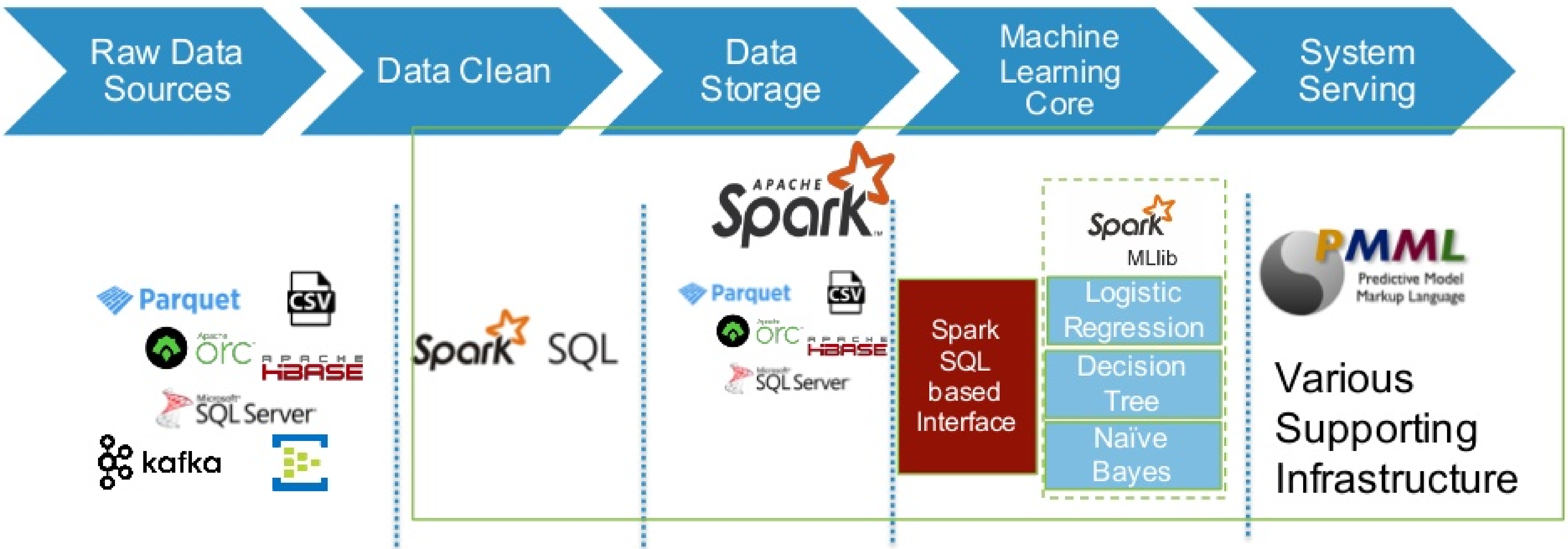
"a mature system might end up being (at most) 5% machine learning code and (at least) 95% glue code"[1]

"Glue code is costly in the long term because it tends to freeze a system to the peculiarities of a specific system"[1]



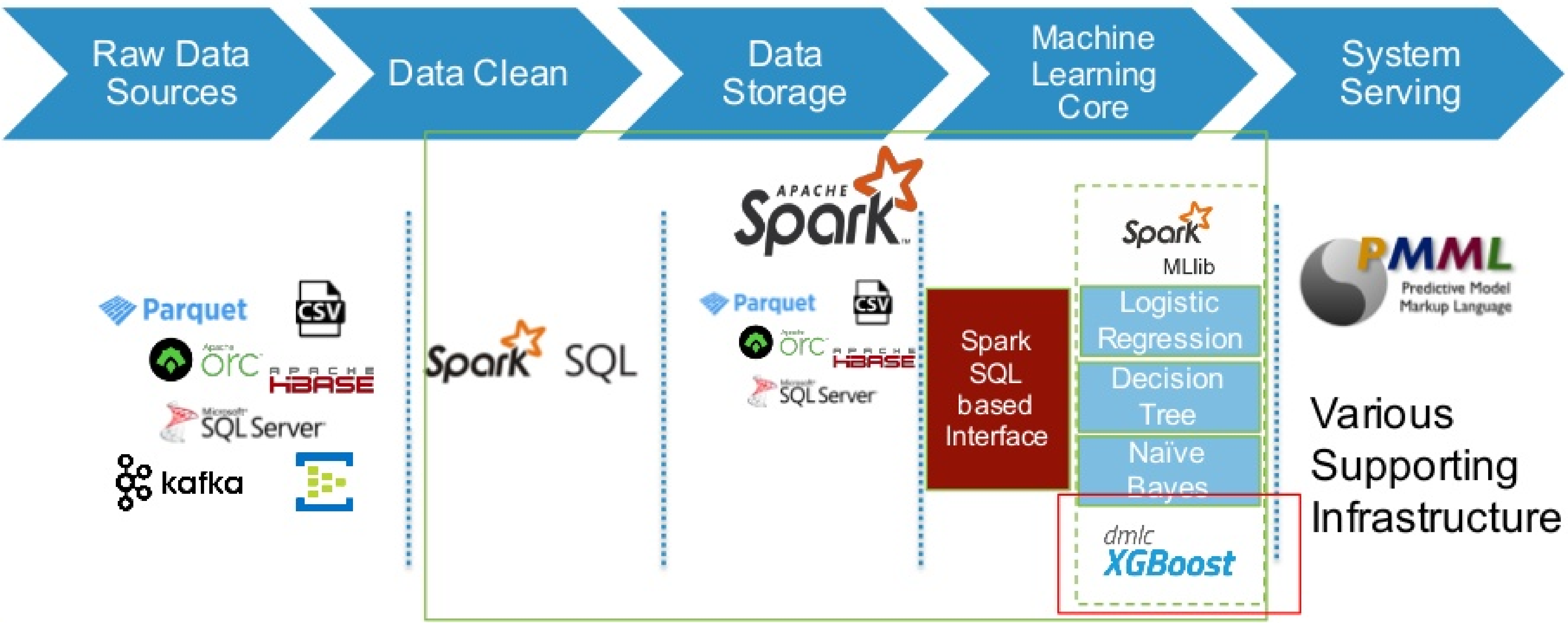
[1] D. Sculley, et al., Hidden Technical Debt in Machine Learning Systems, NIPS 2015

Biggest Advantage of Spark MLLIB



No Additional Glue Code: run in Spark cluster, use standard Data Source API of Spark SQL and existing tuning/feature engineering utils

Ideal Version of Pipeline



Take XGBoost as one of the algorithms in Spark ML

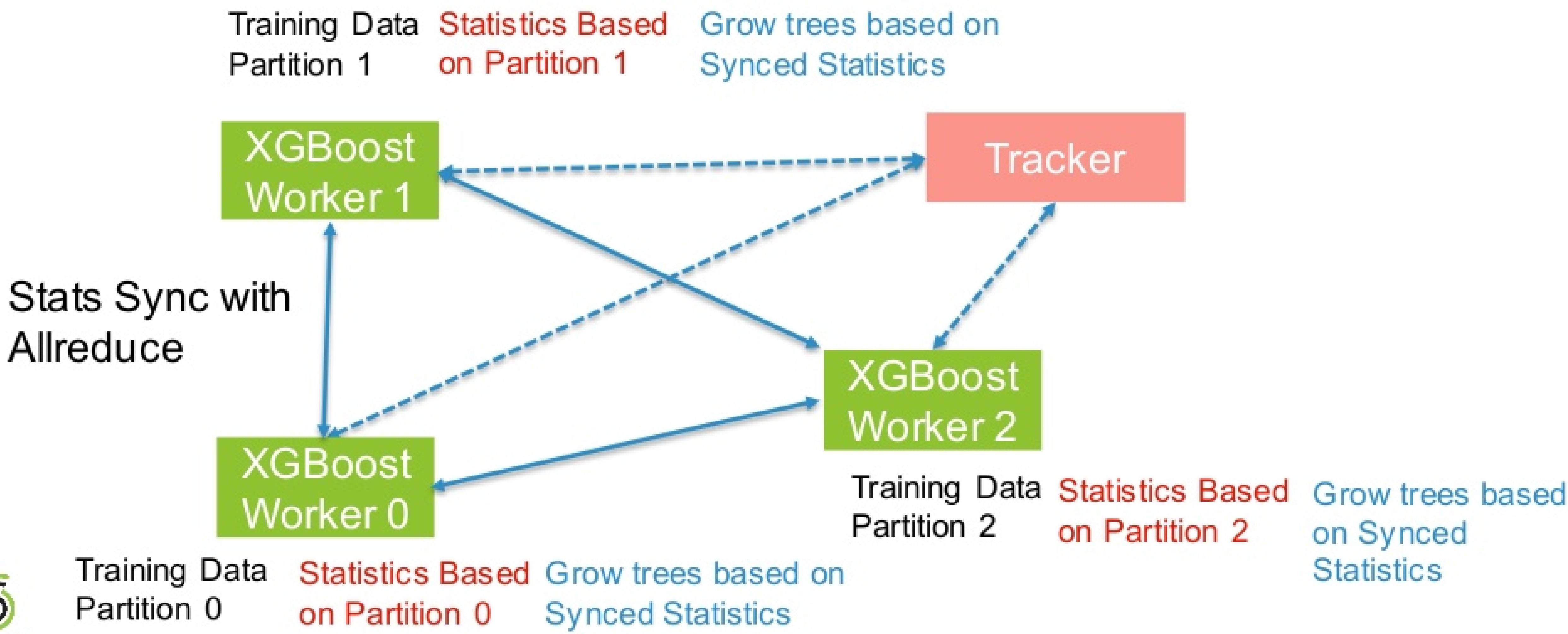
How?

"Apache Spark is an open-source ***Cluster-computing*** framework…centered on a data structure called ***Resilient Distributed Dataset (RDD)***"[1]

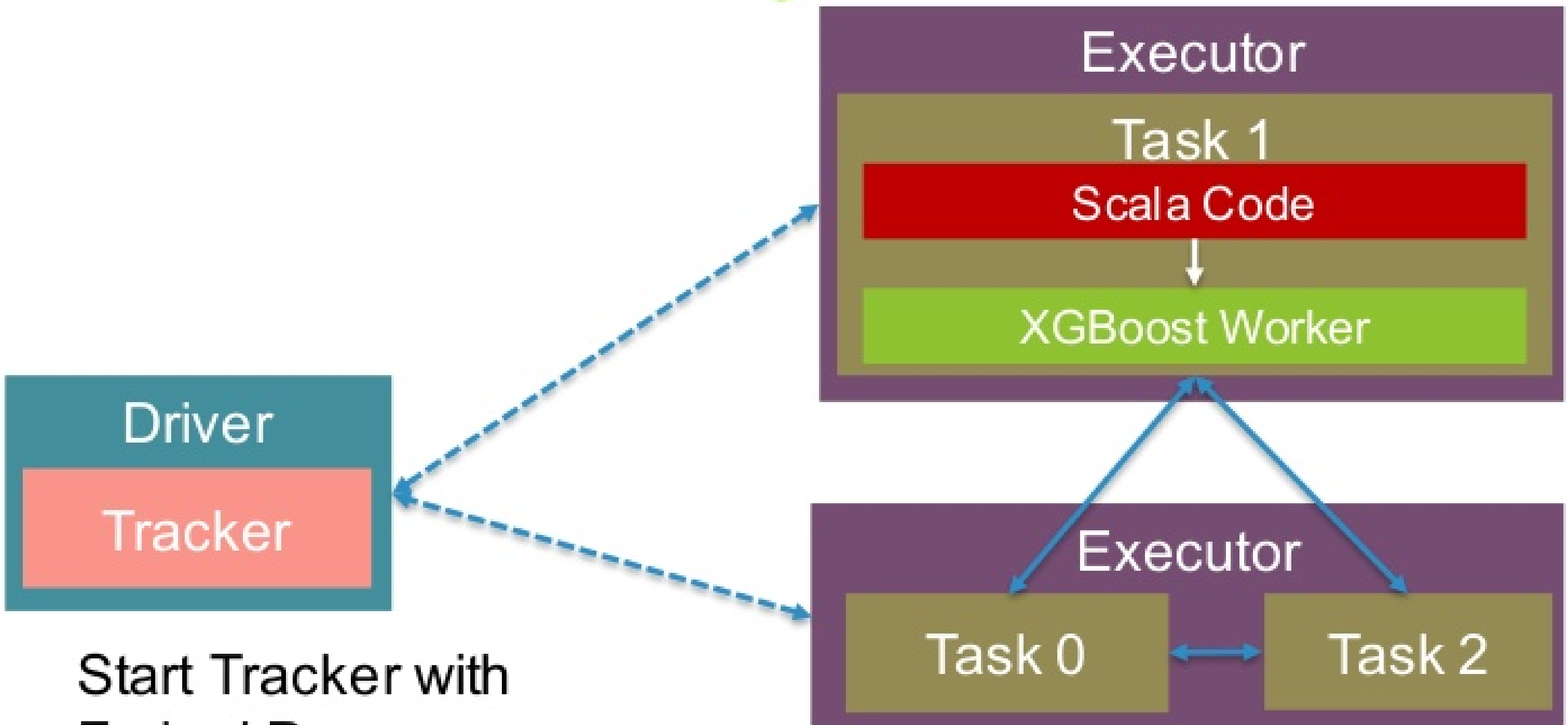
Mission 1: Make XGBoost and Spark Communicate in Execution and Memory Layer

[1] https://en.wikipedia.org/wiki/Apache_Spark

Distributed Training with XGBoost (per Iteration)



Integrate XGBoost and Spark in Execution Layer

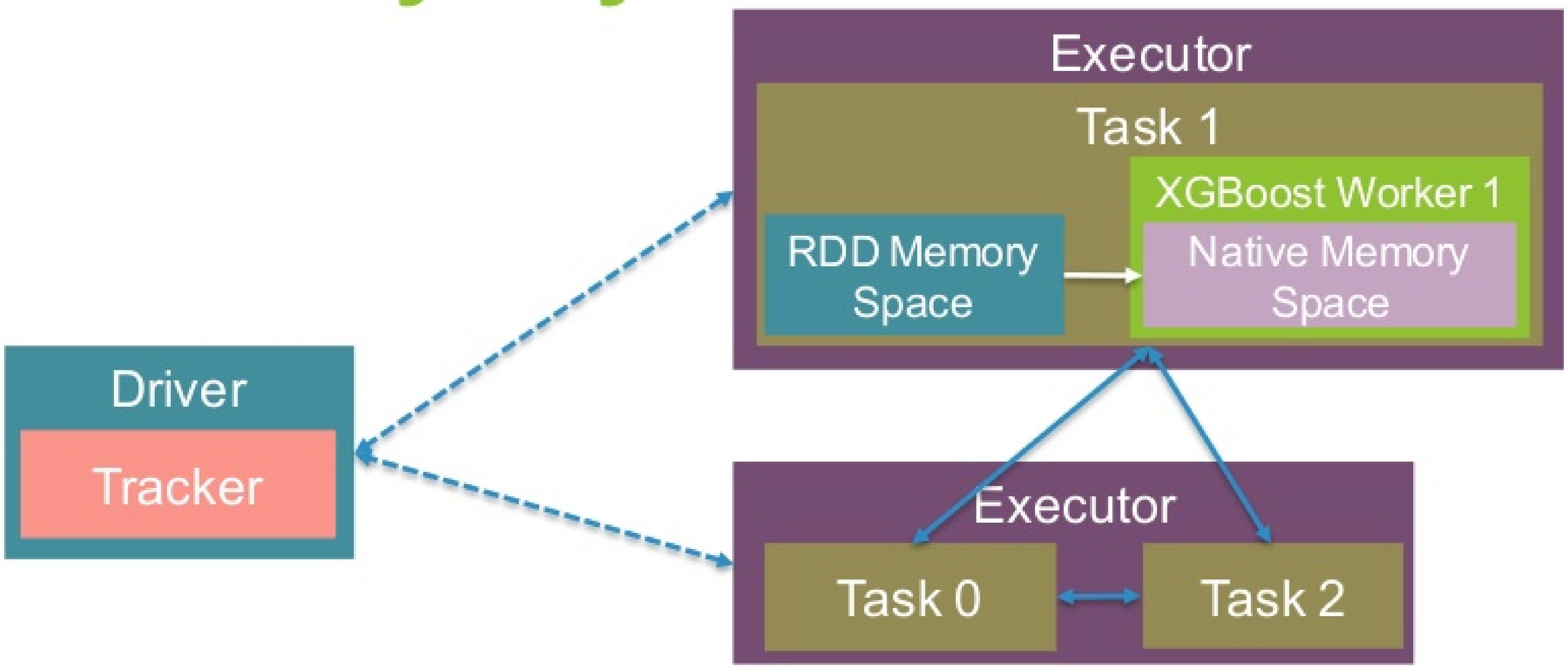


Start Tracker with Forked Process

Integrating Execution Model: Call native libraries through JNI

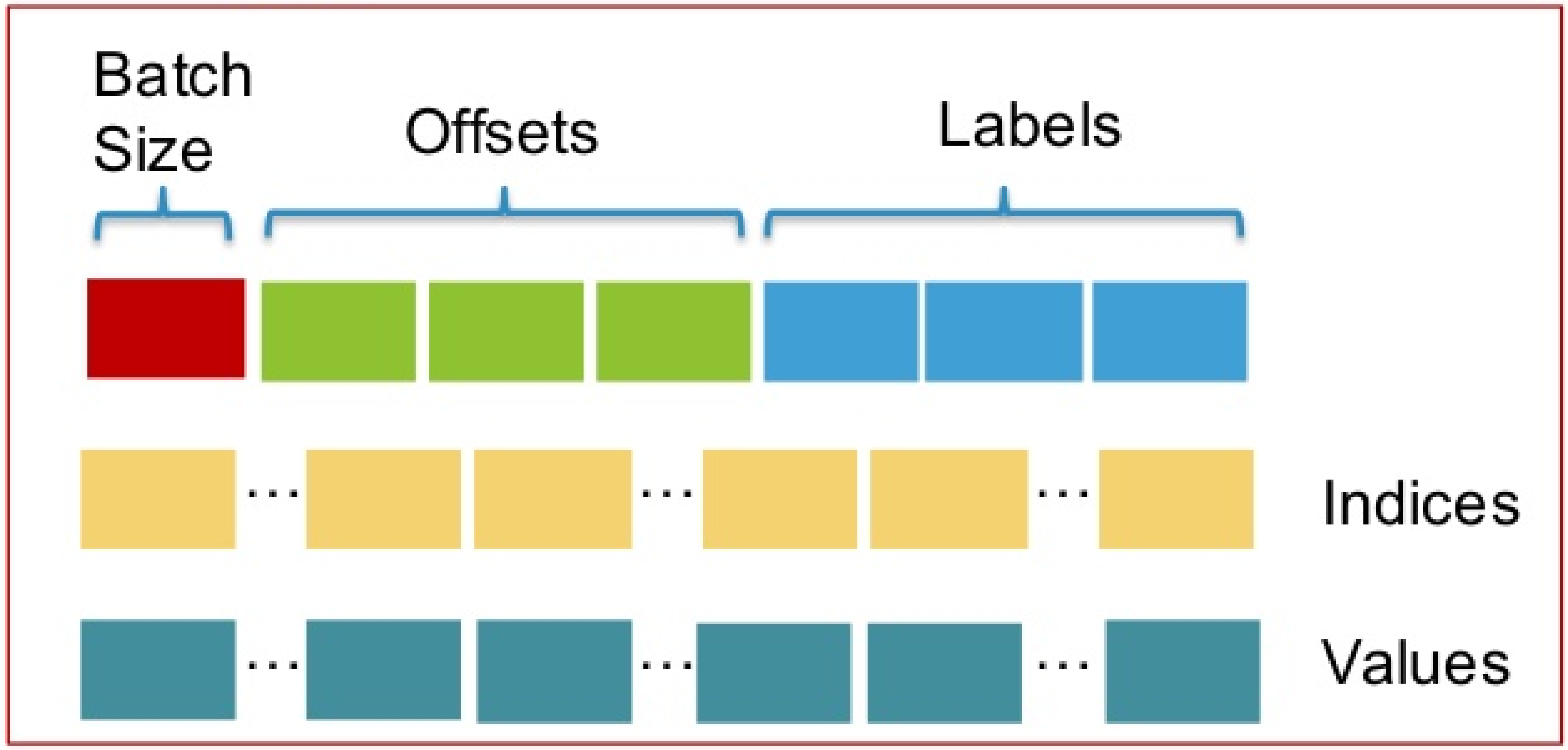
Integrate Distributed Communication: AllReduce Layer (Bypassing shuffle/broadcast)

Integrate XGBoost and Spark In Memory Layer



Connect Memory Layer: Memory copy through JNI

Memory Layout Facilitating Batching Copy



Copy to Native Memory in Batch, interpret by native code

To avoid call JNI for every data point in training dataset

Wrap Internals with APIs

```scala
val trainDF =
sparkSession.read.format("libsvm").load(inputTrainPath)
```

Load Training Data with Spark SQL API

```scala
val paramMap = Map(
  "eta" -> 0.1f,
  "max_depth" -> 2,
  "objective" -> "binary:logistic")
```

Configure XGBoost

```scala
val xgboostModel =
XGBoost.trainWithDataFrame(
  trainDF, paramMap, numRound, nWorkers =
args(1).toInt, useExternalMemory = true)
```

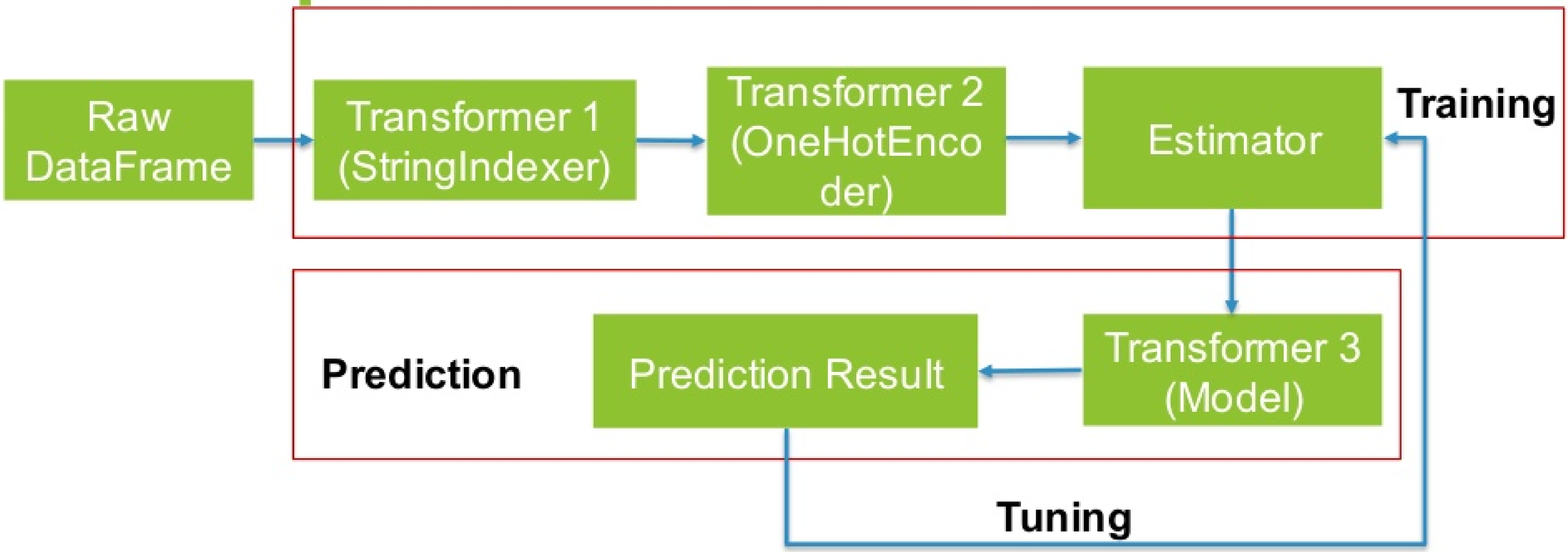
Call XGBoost-Spark API to train

MLlib …make practical machine learning scalable and easy…ML Algorithms… Featurization…Pipelines… Persistence…Utilities…[2]

Mission 2: Integrate with Spark ML Framework

[2] http://spark.apache.org/docs/latest/ml-guide.html

A Machine Learning Pipeline Built with Spark ML Framework



https://dzone.com/articles/distingish-pop-music-from-heavy-metal-using-apache

Fit XGBoost into Spark ML framework

Make XGBoost ***train*** as a native Spark ML Algorithm

XGBoostEstimator

Extends ML's ***Estimator*** triggering distributed XGBoost Workers over training DataFrame

Make XGBoost ***predict*** as a native Spark ML Model

XGBoostModel

Extends ML's ***Transformer***

Make XGBoost be ***tunable*** as a native Spark ML Algorithm

XGBoostParams

Extends ML's ***Parameters*** system

Fitting into Spark ML framework

Extends ML's ***Estimator*** triggering distributed XGBoost Workers over training DataFrame

XGBoostEstimator

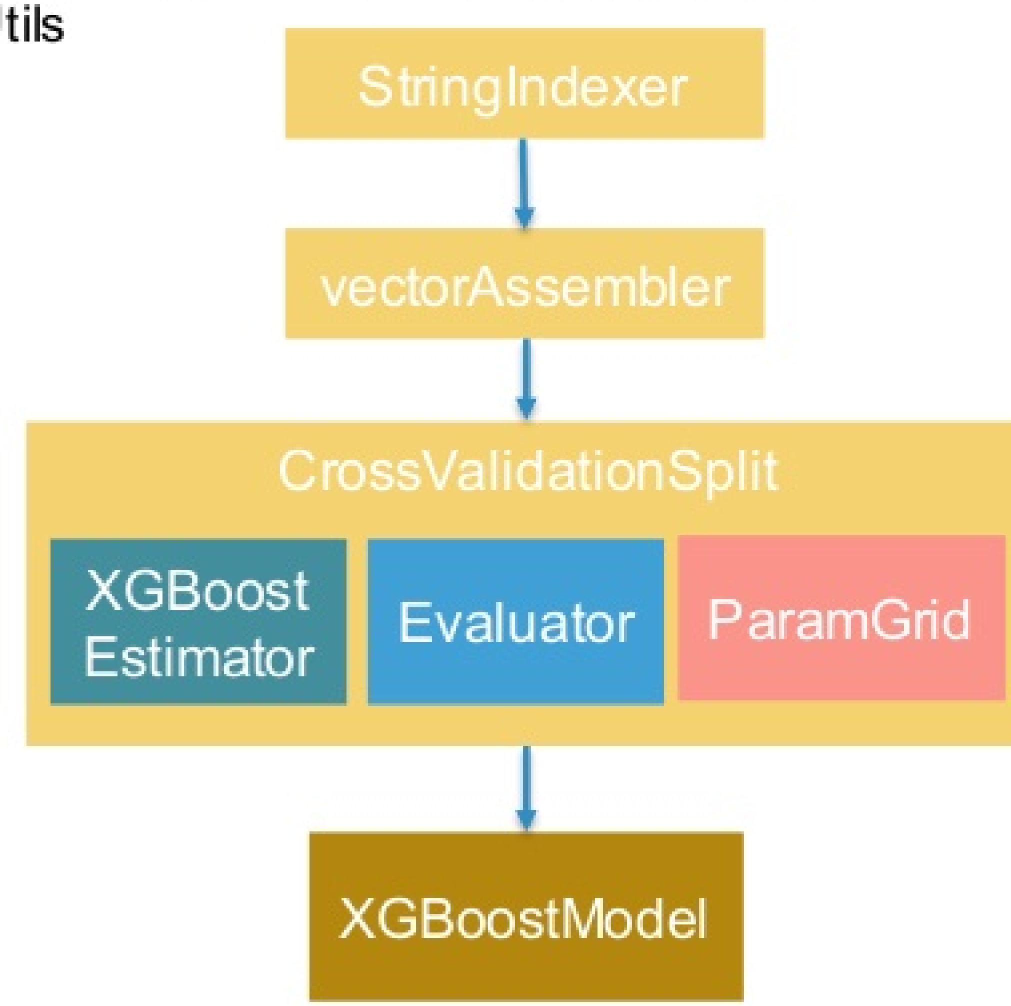
Extends ML's ***Transformer***

XGBoostModel

Extends ML's ***Parameters*** system

XGBoostParams

A Full Pipeline with XGBoost and Spark ML Utils



Building a Unified Pipeline with XGBoost and Spark

```
val pipeline = new Pipeline().setStages(
  Array(monthIndexer, daysOfMonthIndexer, daysOfWeekIndexer,
    uniqueCarrierIndexer, originIndexer, destIndexer, monthEncoder, daysOfMonthEncoder,
    daysOfWeekEncoder, uniqueCarrierEncoder, originEncoder, destEncoder, vectorAssembler))

pipeline.fit(trainingSet).transform(trainingSet).selectExpr(
  "features", "case when dep_delayed_15min = true then 1.0 else 0.0 end as label")
```

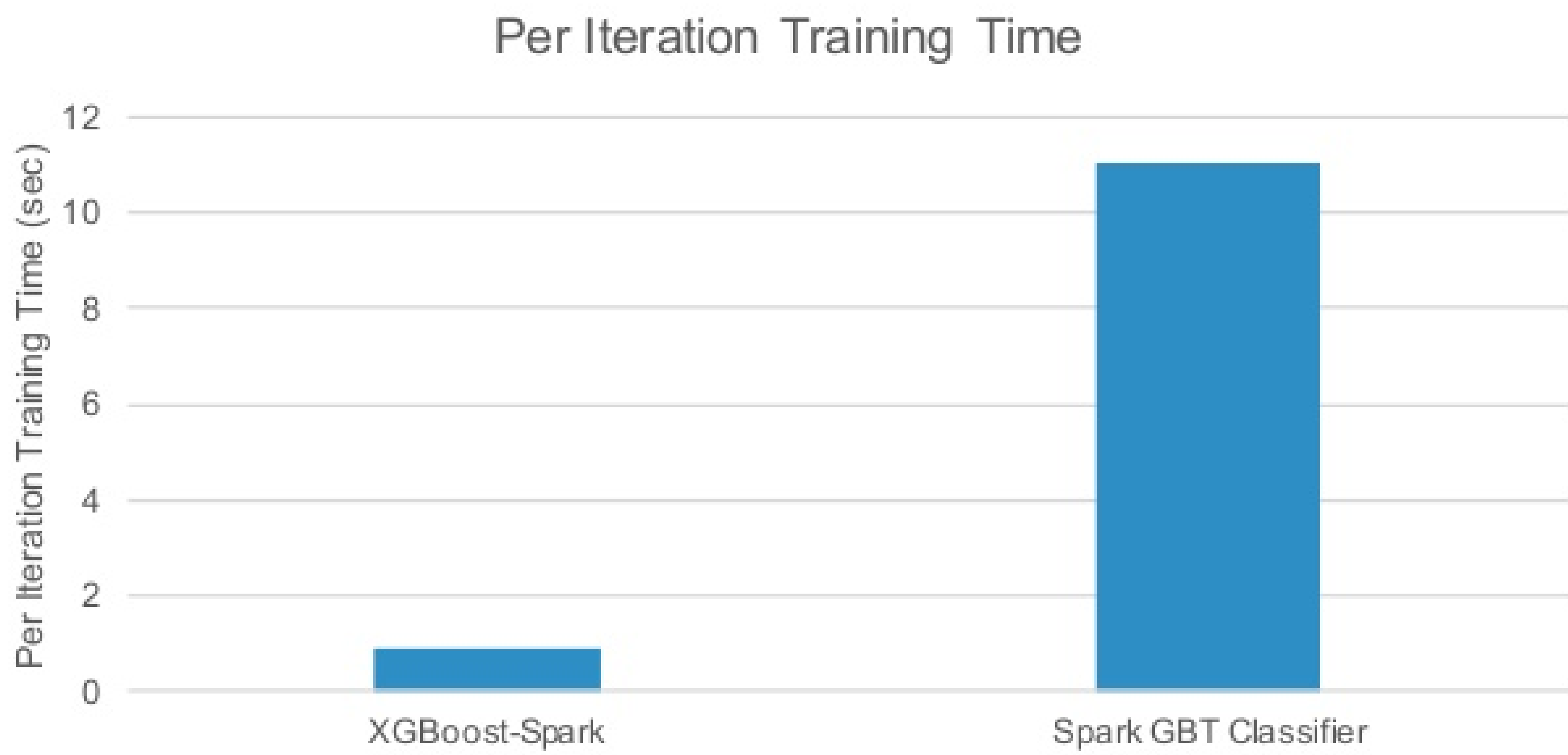
Setting Preprocessing Stages

```
val paramGrid = new ParamGridBuilder()
  .addGrid(xgbEstimator.eta, Utils.fromConfigToParamGrid(conf)(xgbEstimator.eta.name))
  .addGrid(xgbEstimator.maxDepth,
Utils.fromConfigToParamGrid(conf)(xgbEstimator.maxDepth.name).
    map(_.toInt))
  .addGrid(xgbEstimator.gamma, Utils.fromConfigToParamGrid(conf)(xgbEstimator.gamma.name))
  .addGrid(xgbEstimator.lambda, Utils.fromConfigToParamGrid(conf)(xgbEstimator.lambda.name))
  .addGrid(xgbEstimator.colSampleByTree, Utils.fromConfigToParamGrid(conf)(
    xgbEstimator.colSampleByTree.name))
  .addGrid(xgbEstimator.subSample, Utils.fromConfigToParamGrid(conf)(
    xgbEstimator.subSample.name))
  .build()
val cv = new CrossValidator()
  .setEstimator(xgbEstimator)
  .setEvaluator(new BinaryClassificationEvaluator().
    setRawPredictionCol("probabilities").setLabelCol("label"))
  .setEstimatorParamMaps(paramGrid)
  .setNumFolds(5)
val cvModel = cv.fit(trainingSet)
cvModel.bestModel.asInstanceOf[XGBoostModel]
```

Searching Optimal Parameters of XGBoost with CrossValidation

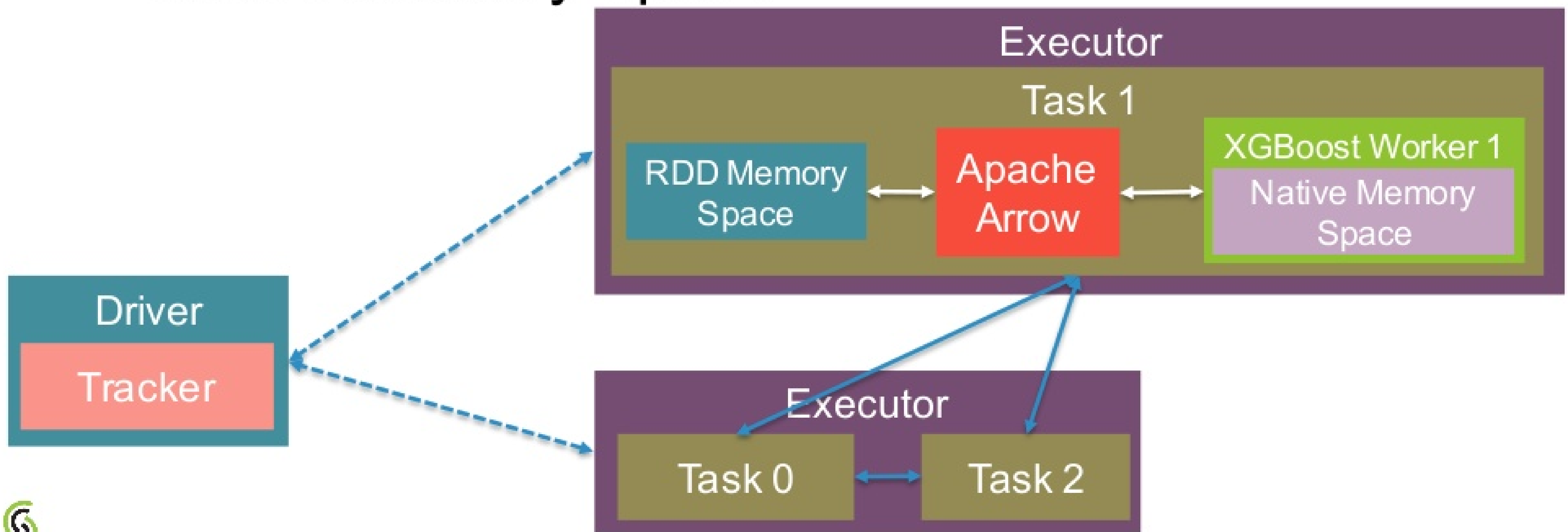
Performance Evaluation

Airline Dataset (22M examples), 48 Workers in XGBoost/Tasks in Spark
Hardware: 6 D4V2 VMs on Azure serving Spark Executors



Future Work

- Unified Memory Space



https://arrow.apache.org/

What we can learn from the design of XGBoost-Spark

- Spark ML framework facilitates us to implement something like XGBoost-Spark
- Beyond the current Spark ML…
  – More pain points in ML pipelines, e.g. entanglement (record system behavior), data dependencies (versioning training dataset), …

Summary

- Introduction to XGBoost & XGBoost Spark
- Machine Learning algorithm is only a very small part of the complete data processing/analytic pipeline
  - Embed XGBoost to Apache Spark ML Pipeline (XGBoost-Spark) to resolve your headaches
- A new view to Spark/Spark ML

Acknowledgement

- Special thanks to Tianqi Chen, who created XGBoost project and offered strong support when I built XGBoost-Spark
- Thanks to XGBoost Committers/Contributors/Users who keep working on improving the project
- Thanks to McGill University which supports me working on the project

Thank You!!!

https://github.com/dmlc/xgboost

https://github.com/dmlc/xgboost/jvm-packages