Apache Spark
for machine learning

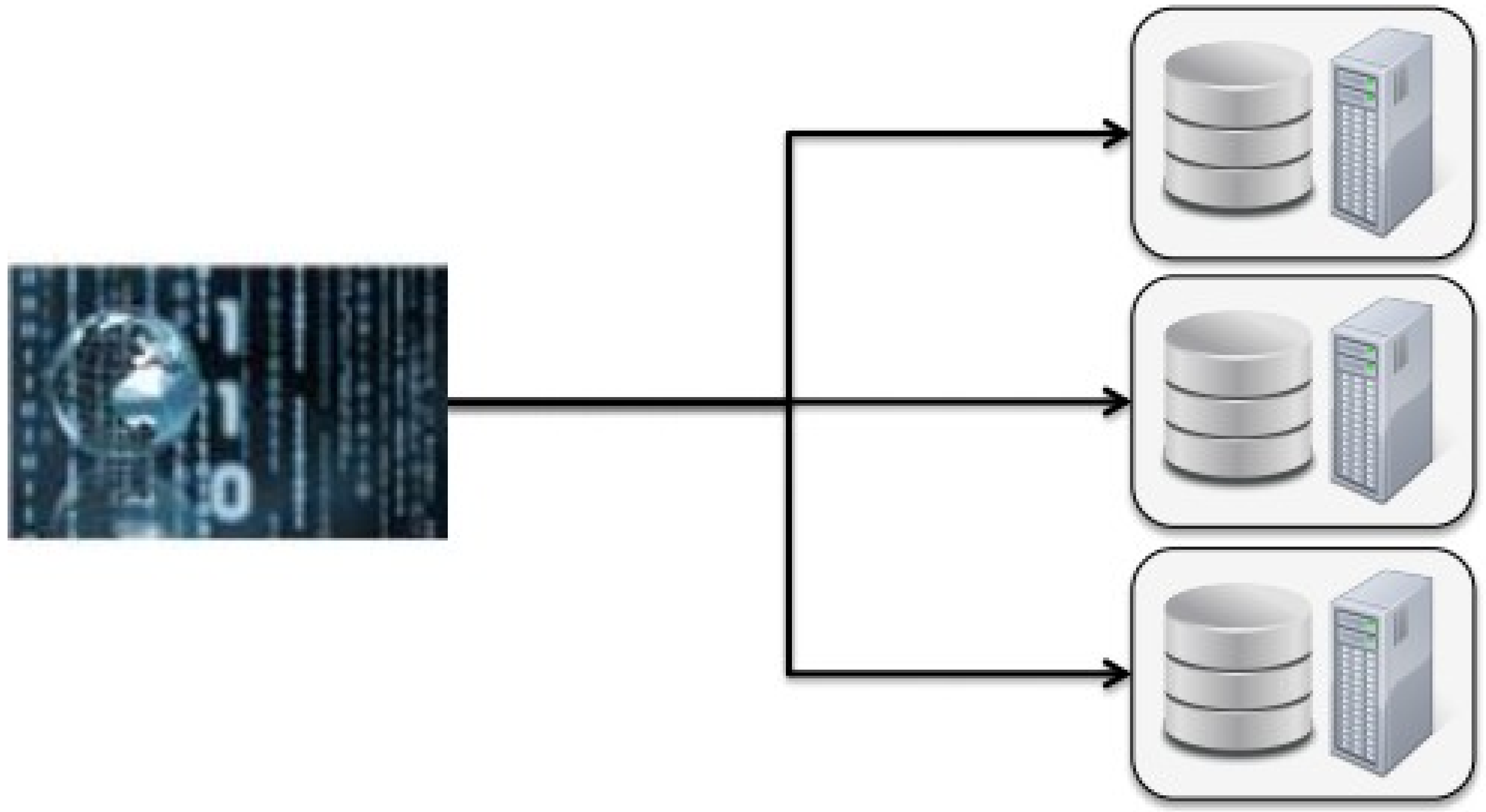
Connectivity · Intelligence · Security
for your business

(+34) 986 120 430 | gradiant@gradiant.org | www.gradiant.org

Quick recap

What's big data, again?

- as data increase becomes the bottleneck
- the solution, more machines
- the data is distributed in several machines
- in this way each machine only process a share of the data



Quick recap

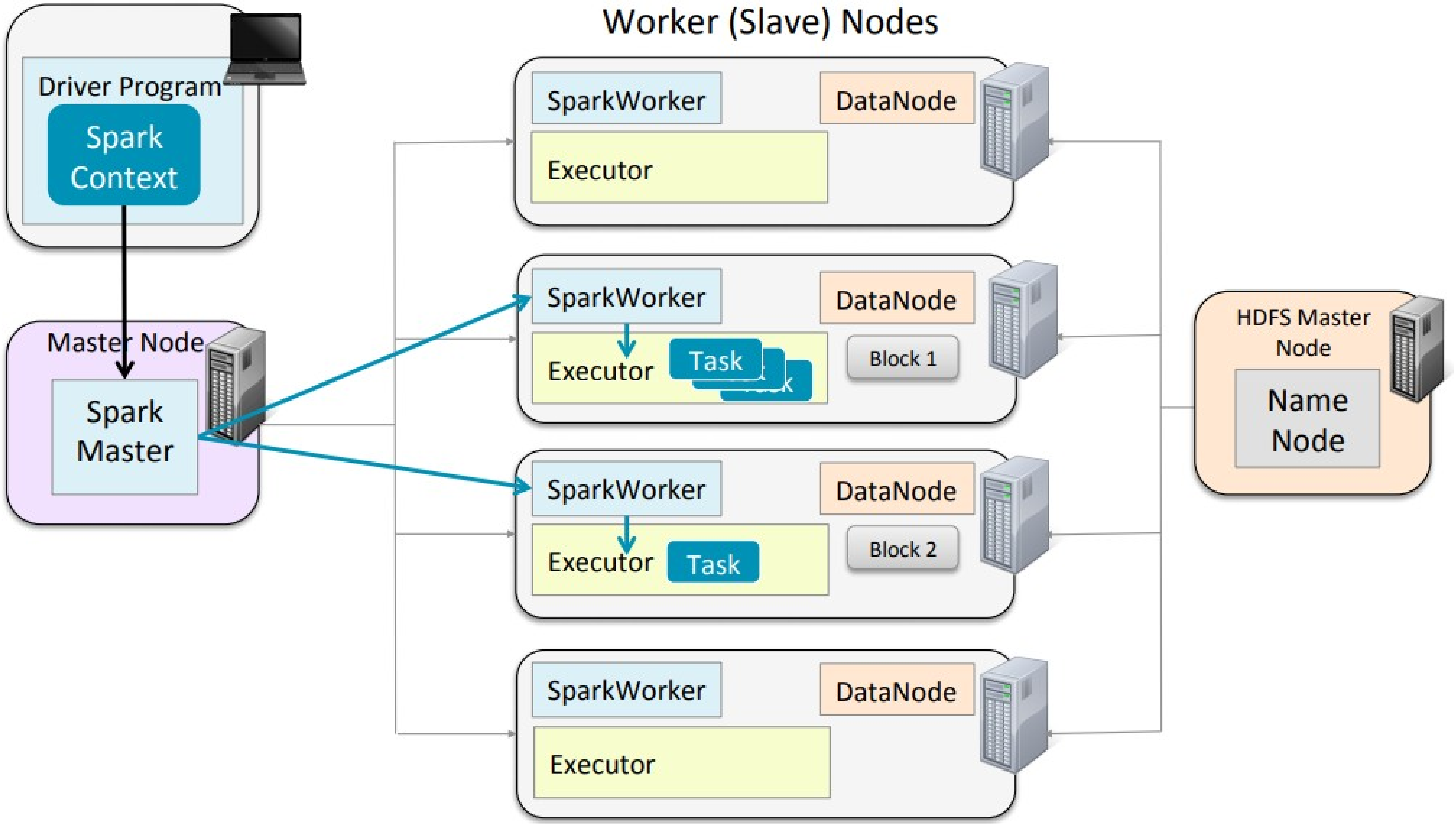
What's spark, again?

Apache Spark is a fast, general engine for large-scale data processing on a cluster

- High level programming framework
- Cluster computing
- Distributed storage
- Data in memory
- Provides fault tolerance
- Adding nodes adds capacity proportionally

Quick recap

What's the spark architecture, again?



Quick recap

What's rdd, dataframe, etc, again?

- RDD Is building block of spark. No matter which abstraction Dataframe or Dataset we use, internally final computation is done on RDDs
    - Distributed: Processed across the cluster
    - Resilient: If data in memory is lost, it can be recreated
    - No schema defining columns and rows
- DataFrame offers huge performance improvement over RDDs
    - Custom Memory management
    - Optimized Execution Plans
    - but Lack of Type Safety
- DataSet is is an extension to Dataframe API
    - comes with OOPs style and developer friendly compile time safety

Spark and ML

MLlib is first Spark scalable machine learning library

MLlib provides main Ml Algorithms like:

- classification
- regression
- clustering
- collaborative filtering

Machine learning problems

- developers use to think that machine learning is basically learning algorithms
- Libraries like MLLIB and Mahout implements this way of thinking
- but productions is different
- in the real world we have end to end applications with several steps
  - data exploration
  - data preparation
  - model training
  - model evaluation
  - model tuning
  - and most important, repeat the process several times

Problems with Spark MLLIb

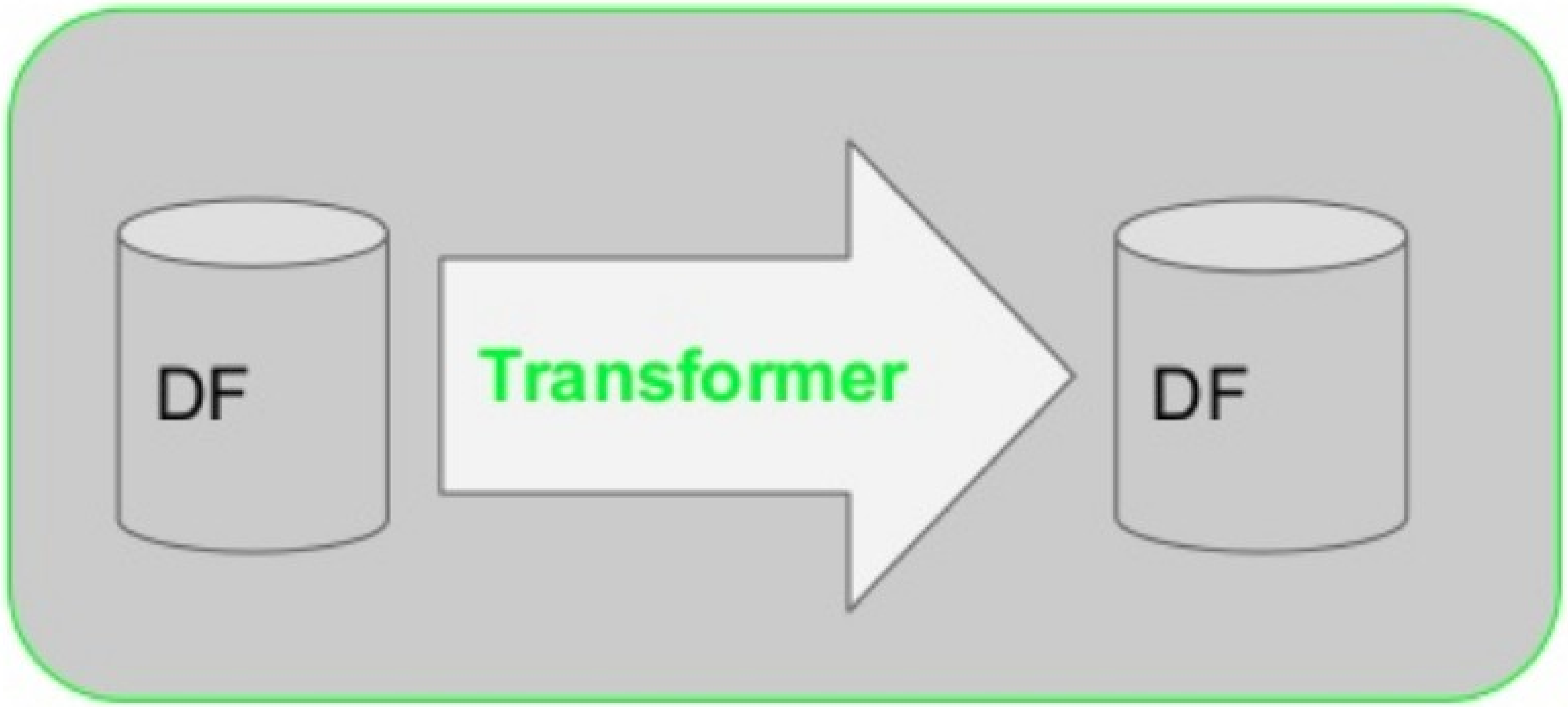
- build on top of RDD
- only focus on model learning
- no way implement a pipeline
- no way combine steps
- no uniform across algorithms

Solution: Spark ML

- provide API for create workflows
- uniform across algorithms
- build on top of dataframes
- MLLib will be deprecated

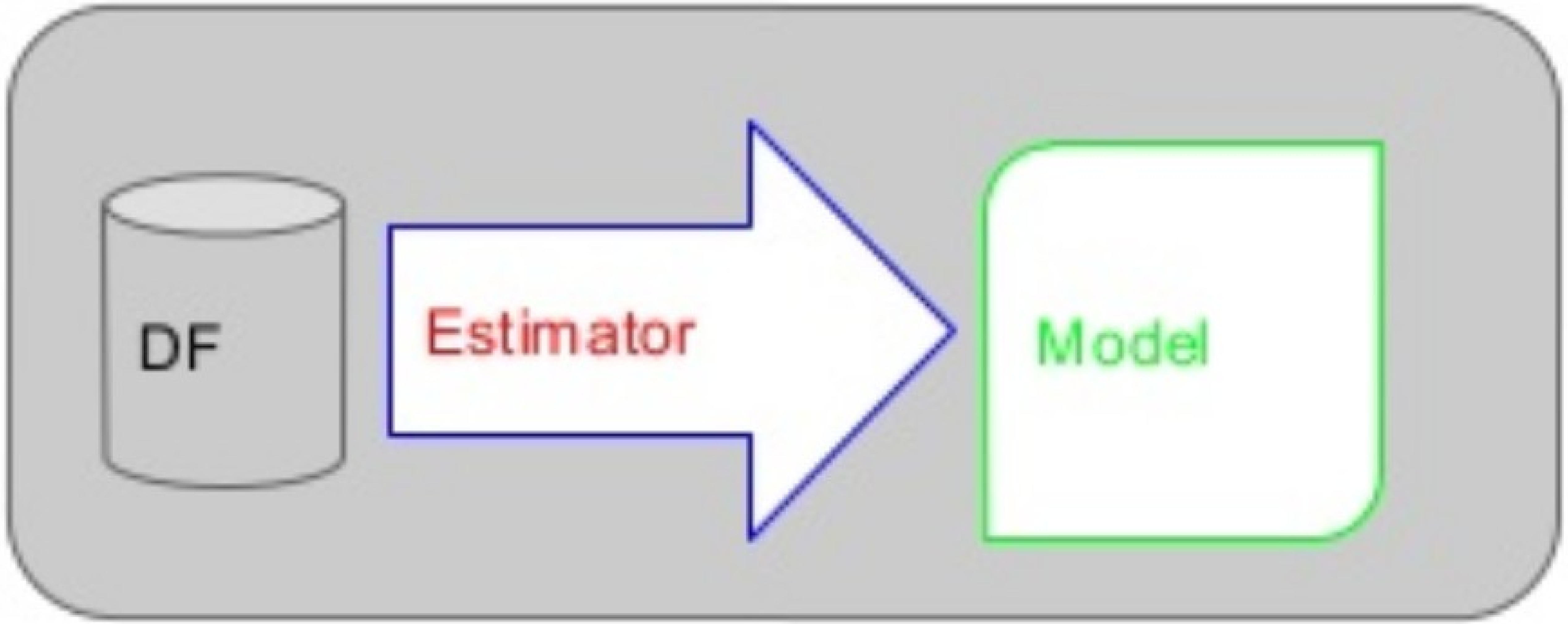
Elements: Transformer

- convert a DataFrame into another DataFrame
- implement the method *transform()*



Elements: Estimator

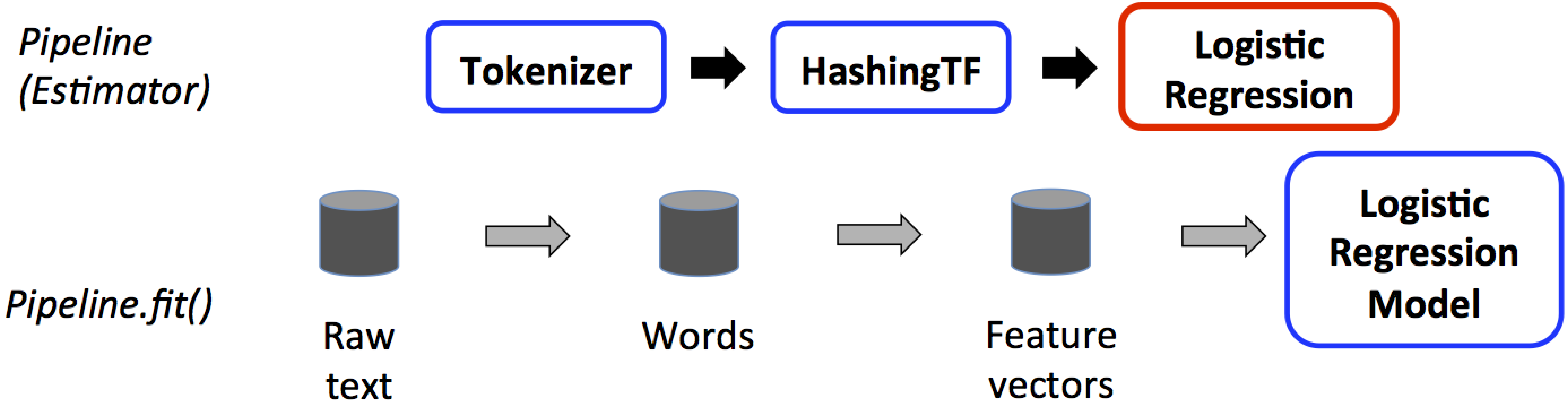
- take a DataFrame and produces a Model
- implements the method *fit()*



Elements: Pipelines(1)

- sequence of stages
- concat Transformers or Estimators
- Pipeline itself is an Estimator
    - fitted on a DataFrame turning into a model
- We can define a pipeline make differents data sets go through the same steps

Elements: Pipelines(2)



How is an algorithm implemented? (1)

Logistic regression definition

FEATURE VECTOR → $x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix}$ $w = \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_d \end{bmatrix}$ ← WEIGHTS

$$\hat{y} = \frac{1}{1 + e^{-(w_1 \cdot x_1 + w_2 \cdot x_2 + \cdots + w_d \cdot x_d)}}$$ ← PREDICTION

LOSS → $$J = -\frac{1}{N} \sum_{i=1}^{N} y^{(i)} log \hat{y}^{(i)} + (1 - y^{(i)}) log(1 - \hat{y}^{(i)})$$

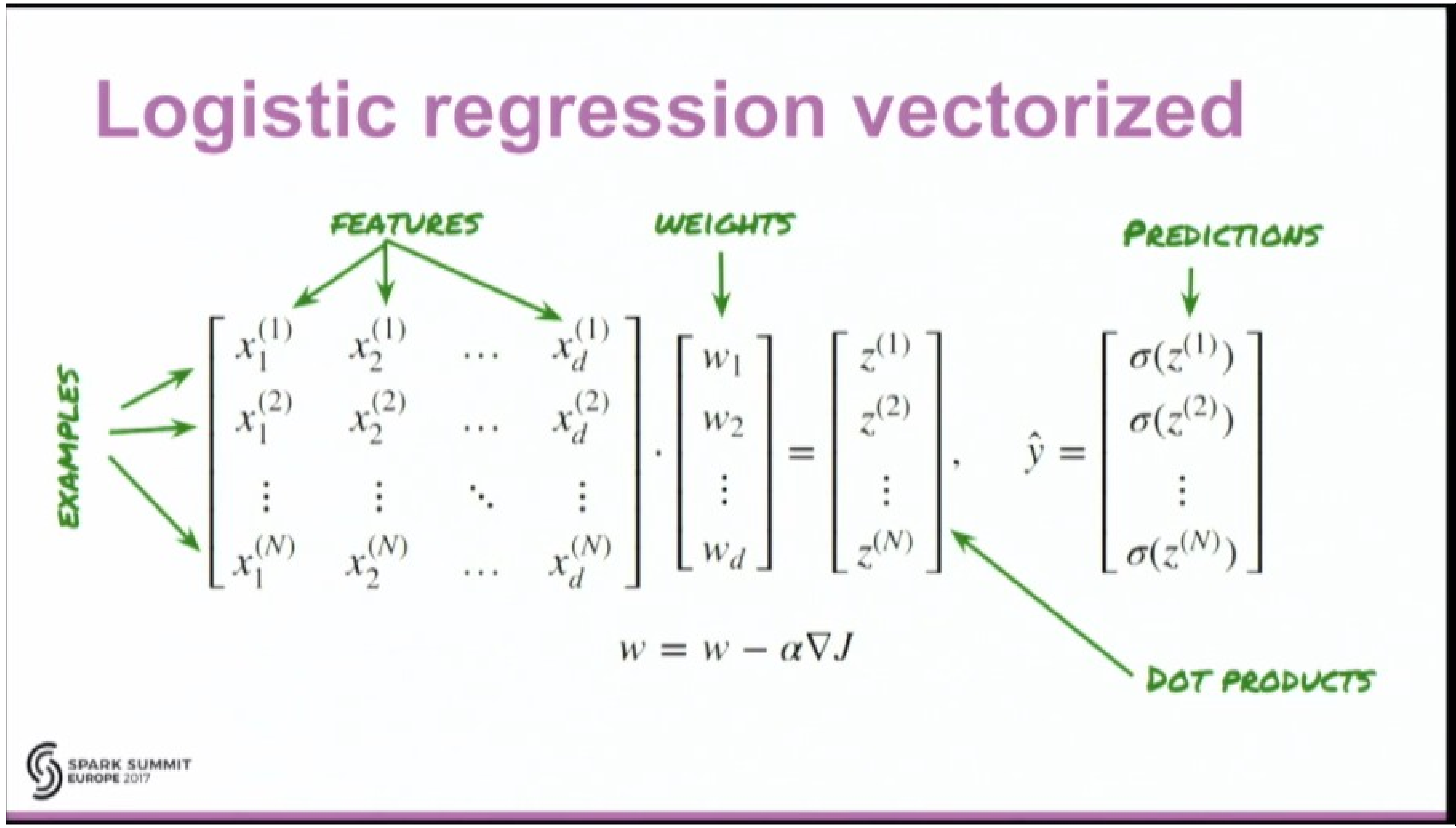
WEIGHT UPDATE

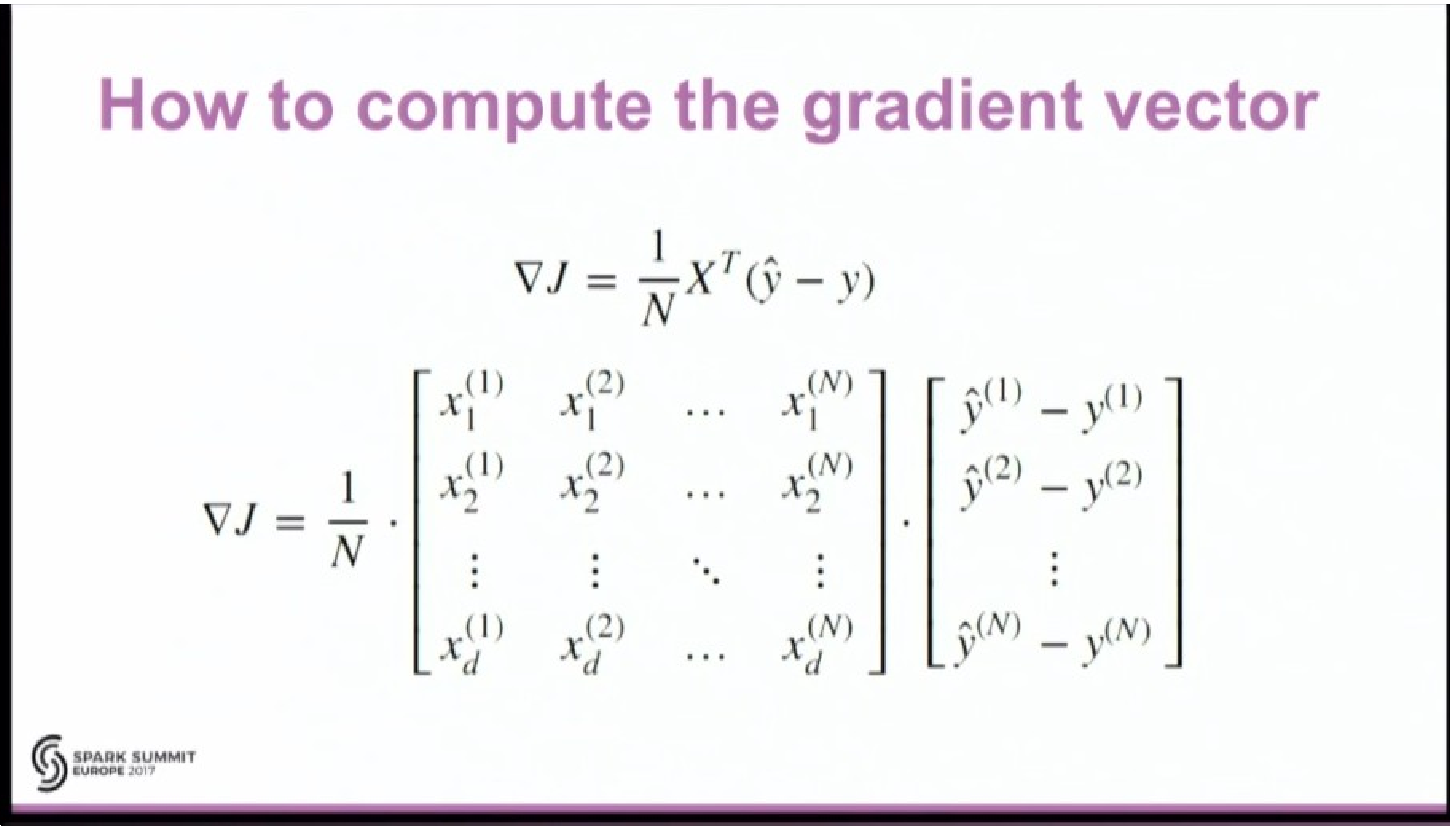
$$w_k = w_k - \alpha \frac{\partial J}{\partial w_k}$$ ← DERIVATIVE OF LOSS

$$w = w - \alpha \nabla J$$ ← GRADIENT

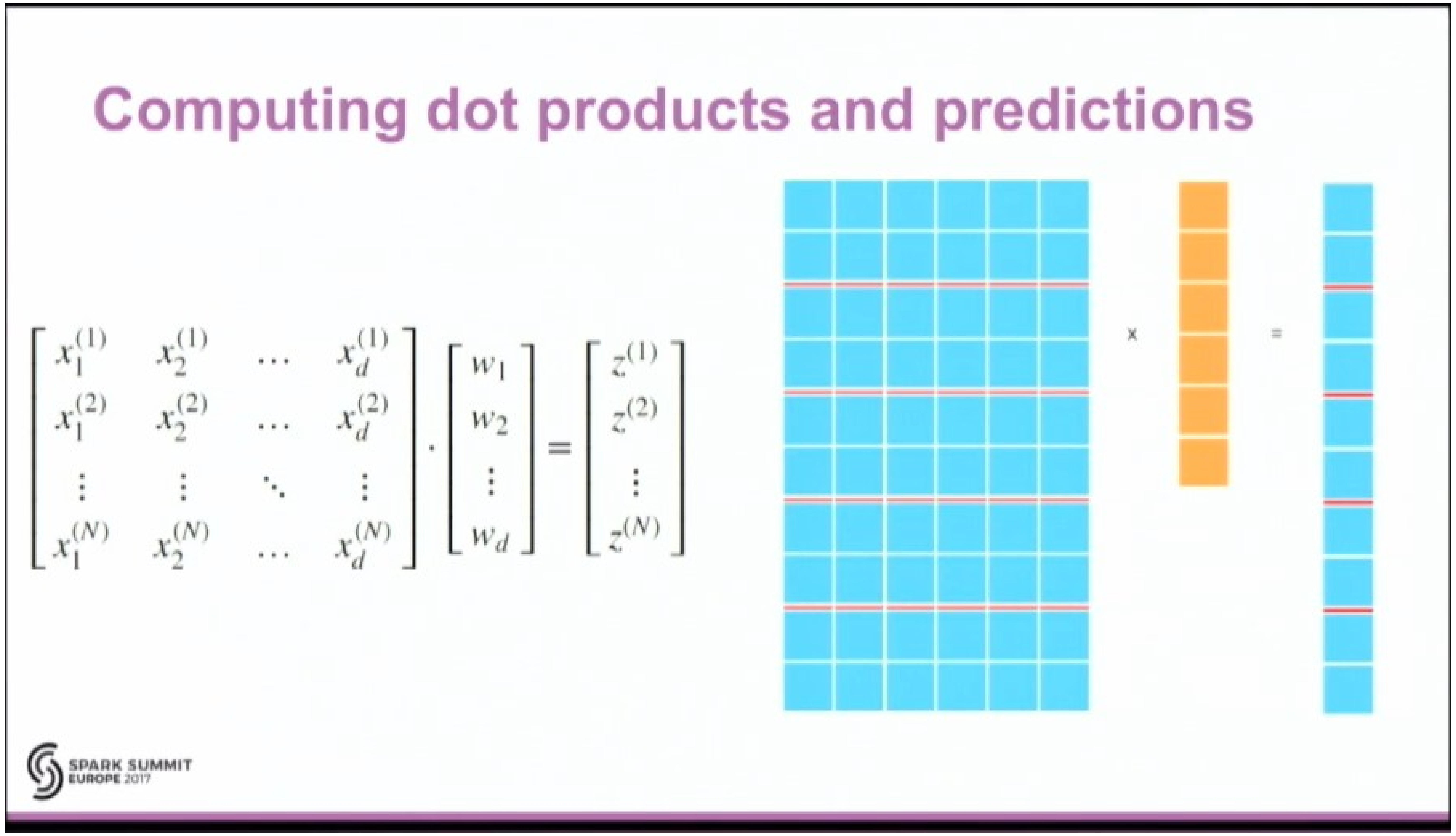
SPARK SUMMIT EUROPE 2017

How is an algorithm implemented? (2)

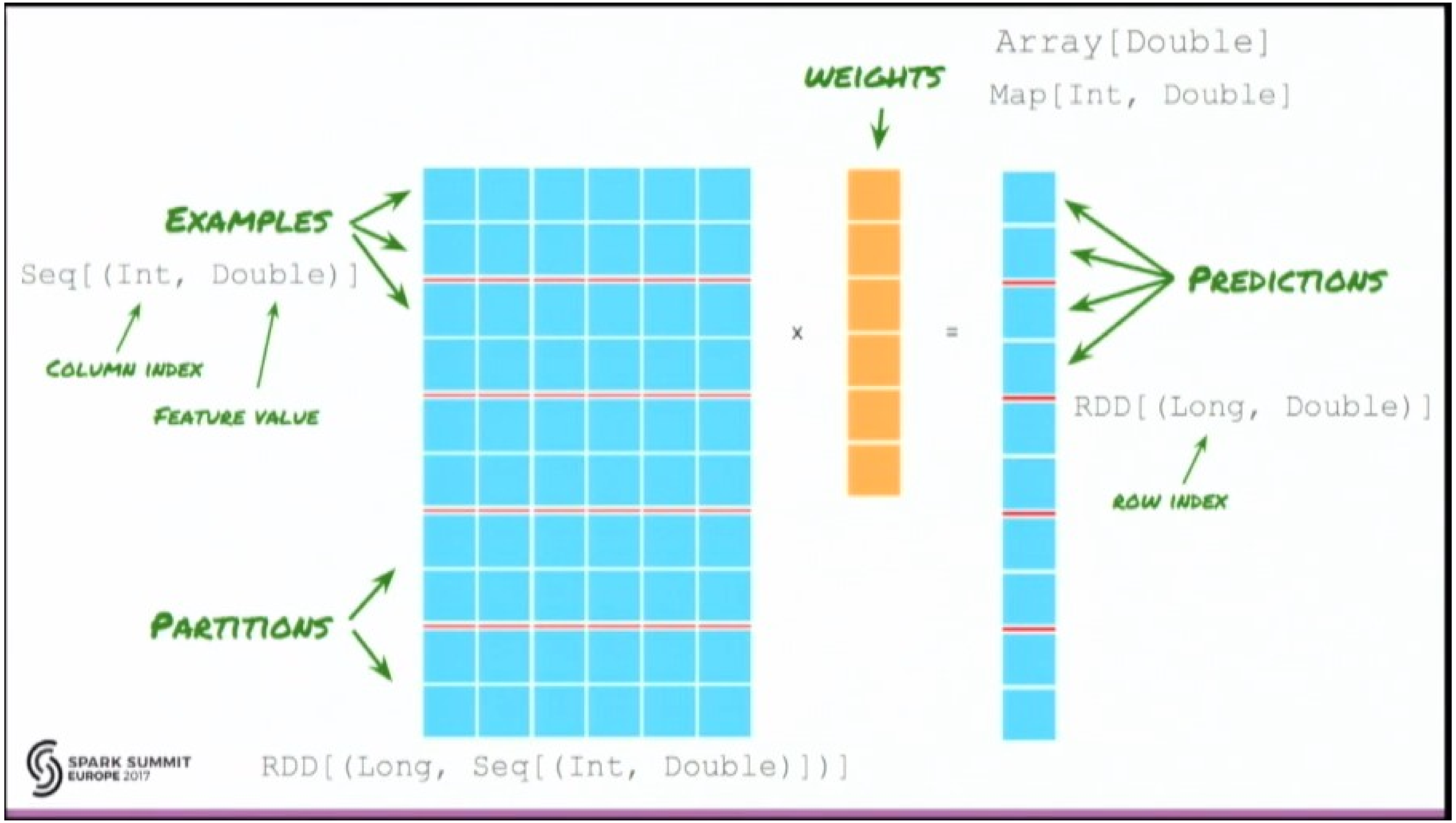




How is an algorithm implemented? (3)



How is an algorithm implemented? (4)




THIS
IS THE
END...