Machine Learning on Big Data

Part Two

DistrictDataLabs

Agenda

- Big Data Approaches
- A Tour of Model Families
- Spark MLlib Algorithms
- Hands-on Lab

Big Data Approaches

Hypothesis One

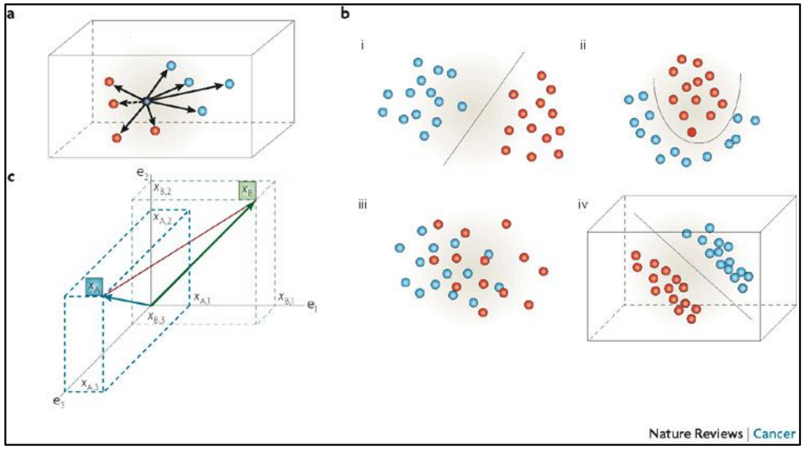
More examples means better machine learning.

Attempt to capture a complete search space.

Hypothesis Two

Increasing dimensions or finding better features improves pattern recognition.

Attempt to divide a search space better.



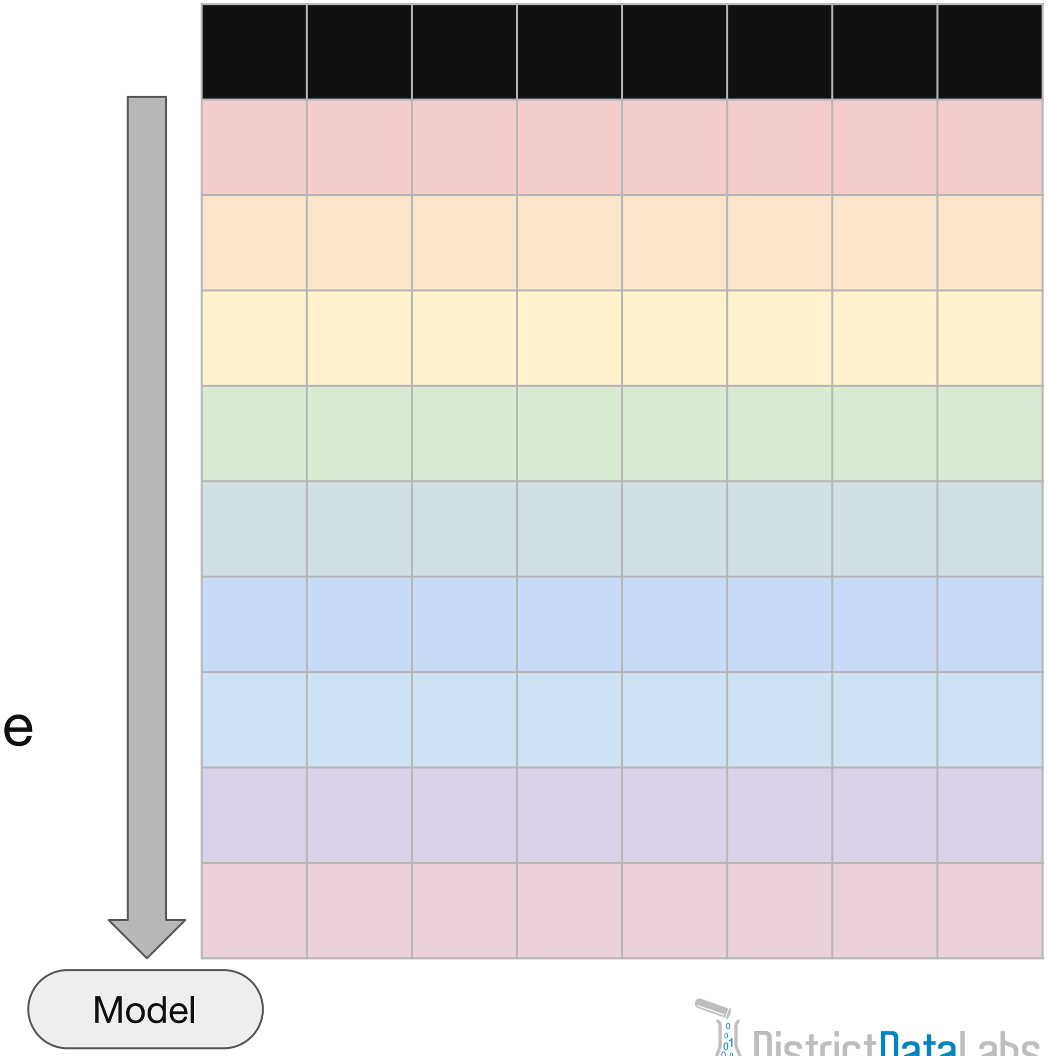
Nature Reviews | Cancer

Sequential Machine Learning

Machine learning generally finds the best set of parameters for a model by optimizing some error function.

(This is most apparent in regression)

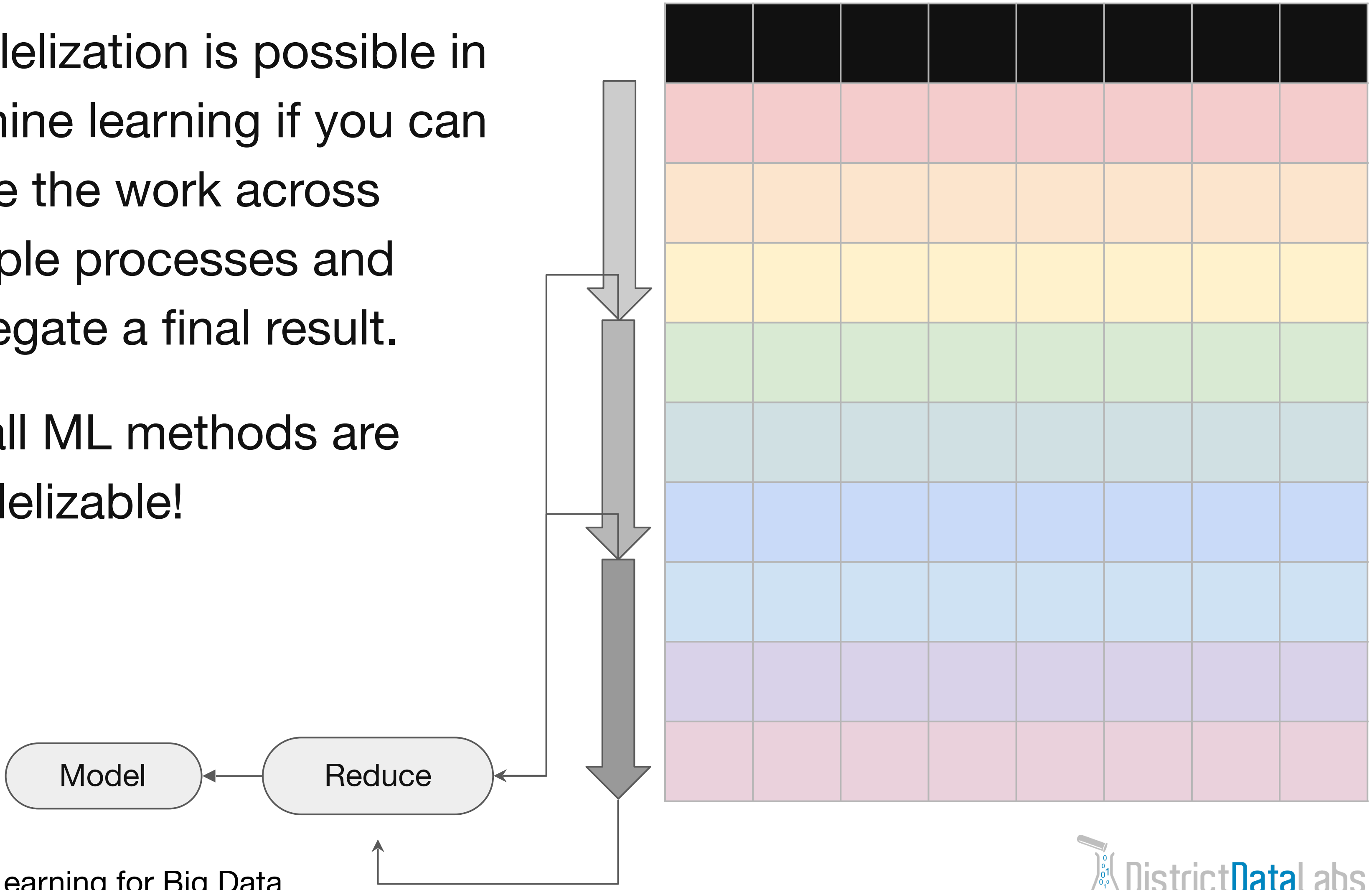
Multiple passes over multiple instances.



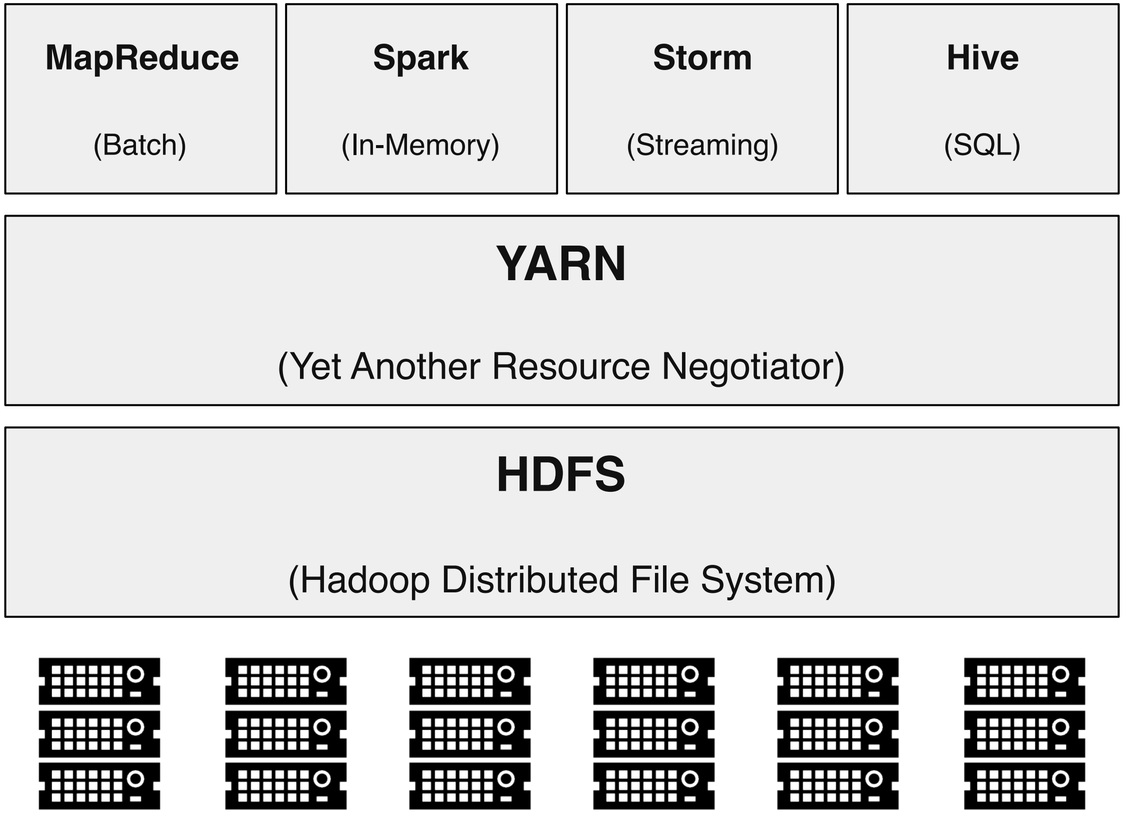
Parallel Machine Learning

Parallelization is possible in machine learning if you can divide the work across multiple processes and aggregate a final result.

Not all ML methods are parallelizable!



Hadoop Stack



MapReduce (Batch)

Spark (In-Memory)

Storm (Streaming)

Hive (SQL)

YARN

(Yet Another Resource Negotiator)

HDFS

(Hadoop Distributed File System)

Cluster of economy disks and processors

Hadoop MapReduce

- Processes structured and unstructured data stored in HDFS
- Designed to process a large volume of data
- Batch (only) processing - other applications provide additional capability (Storm, Impala, Hive, etc.)
- Cannot handle real-time data
- Reads/writes from disk => slower computation
- Every operation needs to be hand-coded.
- High latency
- Non-interactive
- Paradigm shift for programming

Hadoop MapReduce

- Requires an external job scheduler for complex flows (Oozie)
- Resilient to system faults or failures.
- Fault-tolerant
- Highly-scalable
- Basic data processing engine
- No in-memory caching

Spark

- Handles batch, interactive, interactive and streaming.
- Runs 100x faster in memory and 10x faster on disk (reduced read/write cycles)
- RDDs and DataFrames make data easier to work with
- Process real-time streams in the millions of events per second.
- Low-latency
- Acts as its own flow scheduler
- Fault-tolerant
- Require a lot of RAM
- Data analytics as well as data processing

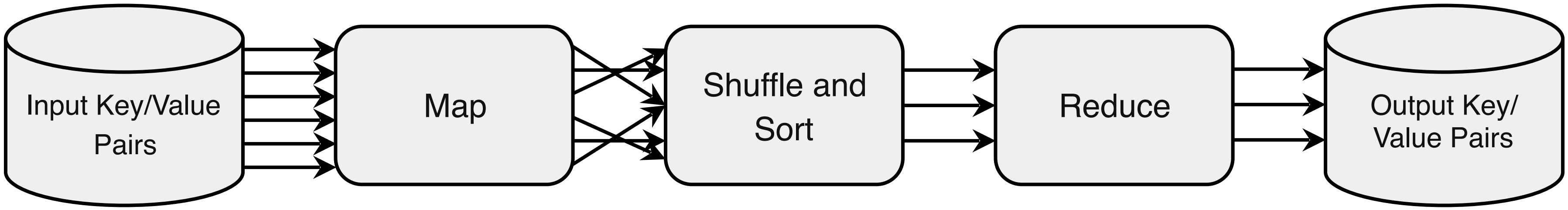
Spark

- Can run SQL queries using Spark SQL
- Highly-scalable (horizontal)
- Machine learning with MLlib
- In-memory caching

Single Map and Reduce

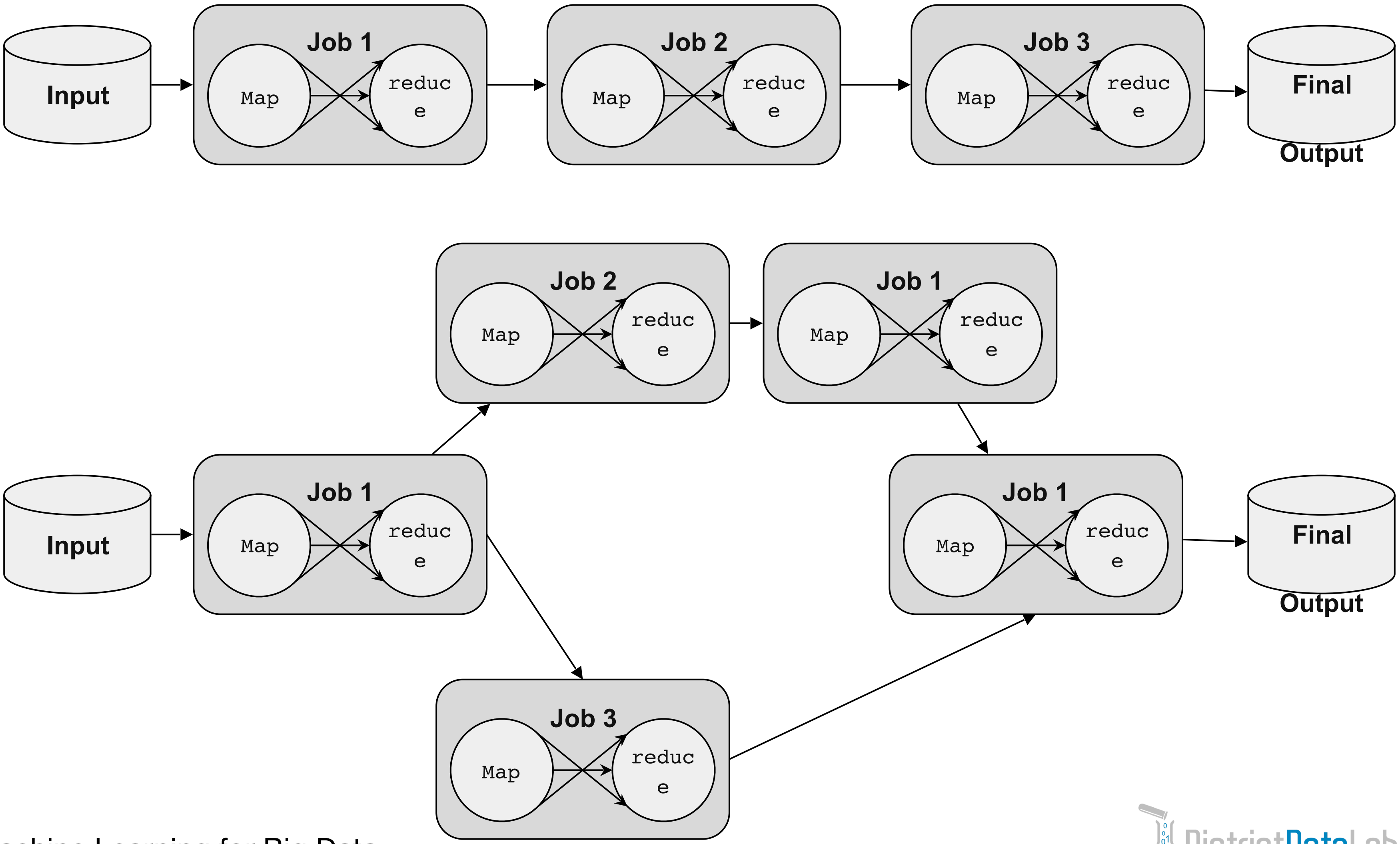
Data is already split into blocks and passed to mapping computations.

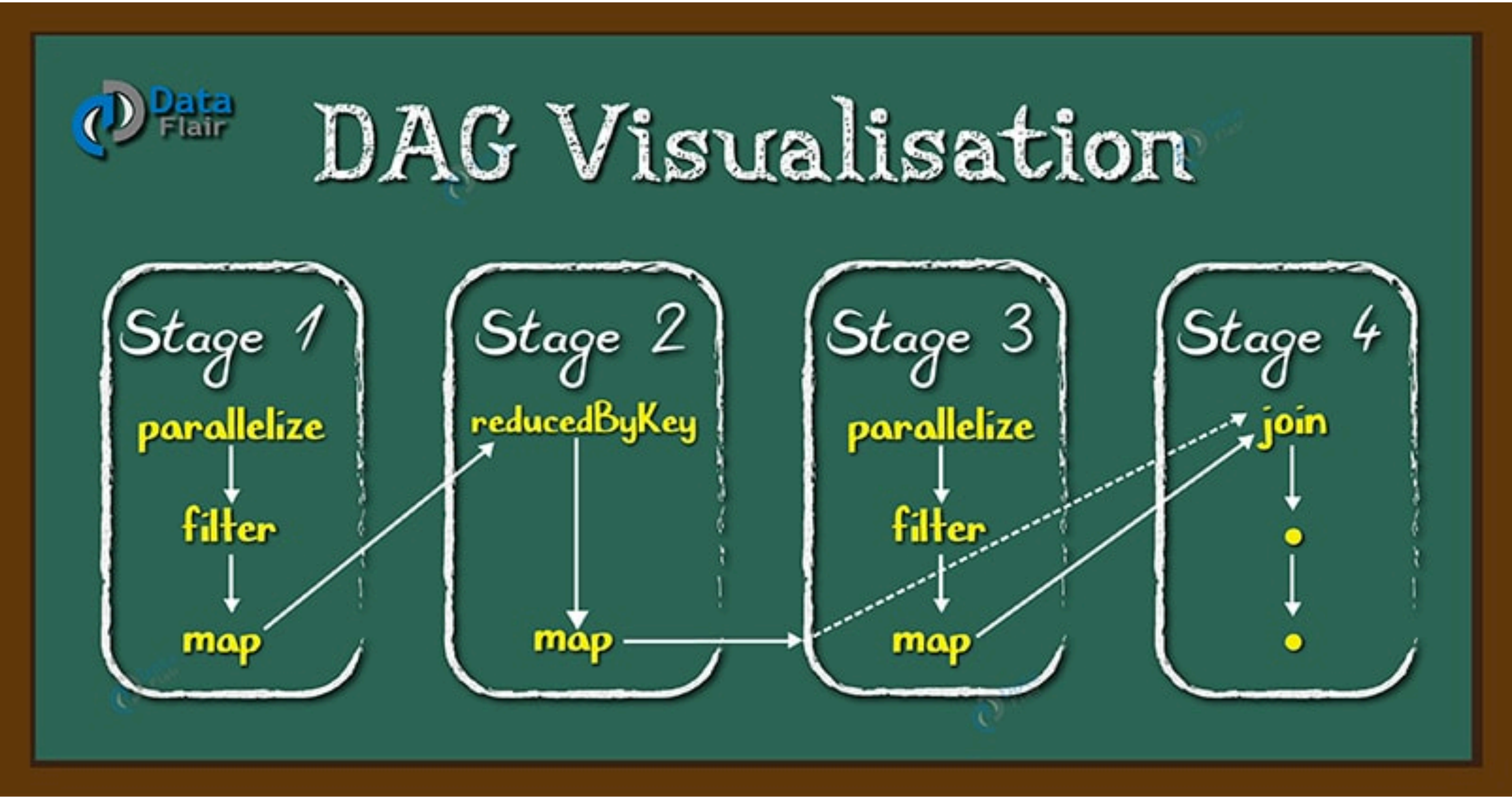
Results of mapping have to be aggregated into a final computation.



Data has to be moved from the mapping tasks to the reducing ones.

Job Chaining for Iteration: Directed Acyclic Graphs





Data Flair
DAG Visualisation
Stage 1
parallelize
filter
map
Stage 2
reducedByKey
map
Stage 3
parallelize
filter
map
Stage 4
join

Directed Acyclic Graph (DAG)

Directed = the connections between the nodes (edges) have a direction: A -> B is not the same as B -> A

Acyclic = "non-circular" = moving from node to node by following the edges, you will never encounter the same node for the second time.

Graph = structure consisting of nodes, that are connected to each other with edges

Directed Acyclic Graph (DAG)

Represents connectivity and causality.

Computations in each job depend on each other.

If computations can be done in parallel but each computation has a maximum execution time, you can calculate the maximum execution time of the entire set => how long will my job take to run?

Ensures computations are performed in order.

Challenges to Distributed ML

Rewriting Algorithms

How do we restructure algorithms to take advantage of parallelism?

E.g. use gradient descent over ordinary least squares.

Iterative Analysis

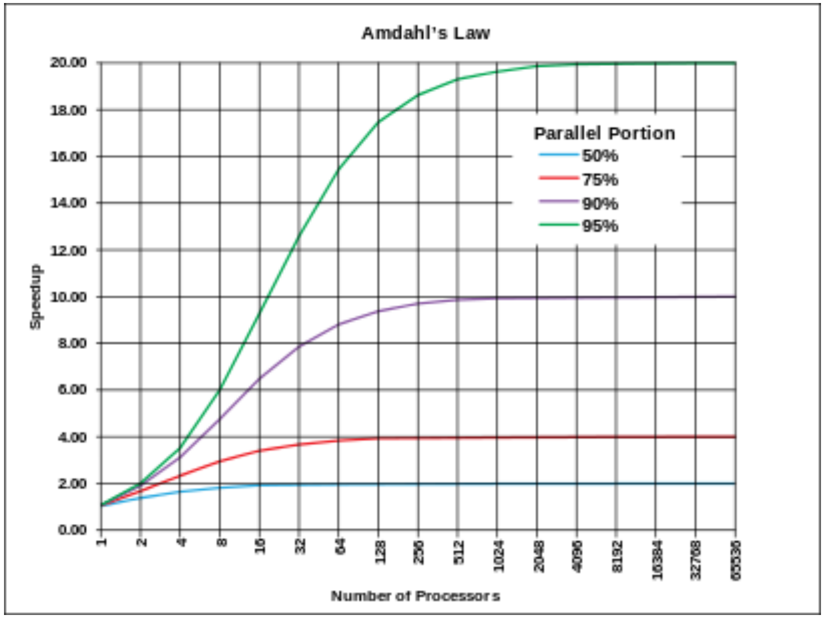
MapReduce writes to disk between Map and Reduce phase.

MapReduce also fixes the number of jobs that can be run.

Amdahl's Law

The amount of speed-up of a parallel system by adding more processors is limited by the percent of the program that is parallelizable.

Parallel processes have setup overhead (sequential portions).



Two Approaches to Big Data ML

Decompose to Memory

Use decomposition methods to reduce input domain into something that can fit into memory (filtering, sampling, summarization, indexing).

Compute model in memory on a single machine (128 GB).

Evaluate model on cluster.

Boost Weaker Models

Use distributed implementations for models that are in Spark to perform computation.

Boost or bag the models together to produce a stronger model.

Performs better with some models, e.g. Random Forest.

Two Approaches to Big Data ML

Decompose to Memory

Use decomposition methods to reduce input domain into something that can fit into memory (filtering, sampling, summarization, indexing).

Compute model in memory on a single node ([illegible]).

Evaluate model on cluster.

Boost Weaker Models

Use distributed implementations for models that are in Mahout or Spark to perform computation.

Boost or bag the models together to produce a stronger model.

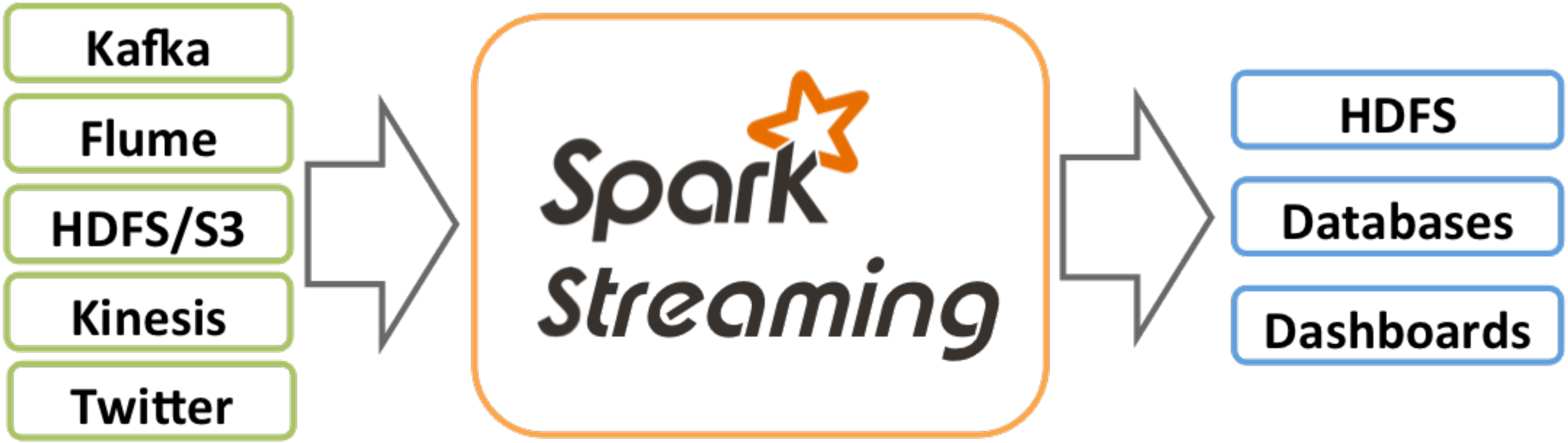
Performs better with some models, e.g. Random Forest.

Using Sklearn in Spark

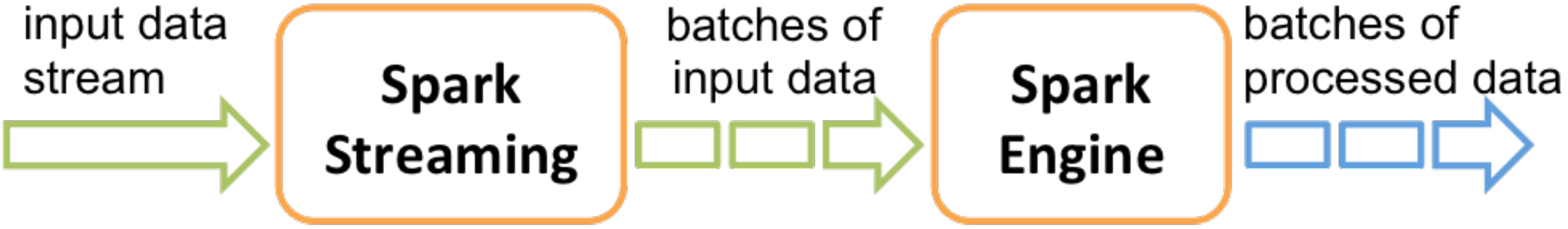
Create a spark job that:

1. Prepares the data - Spark
2. Creates a sample (or filter/summarize/index) - Spark
3. Trains the model on the smaller dataset - Sklearn
4. Validates the model via testing on a test dataset - Spark
5. Applies the model to new observations - Spark

Spark Streaming



Spark Streaming



input data stream

Spark Streaming

batches of input data

Spark Engine

batches of processed data

Spark Streaming

Use the same code for batch processing, streaming or running ad-hoc queries.

Run in standalone or cluster mode.

Uses ZooKeeper and HDFS for high availability.

Key Concepts in Spark MLlib

Spark ML standardizes APIs for machine learning algorithms to make it easier to combine multiple algorithms into a single pipeline, or workflow.

- Transformer
- Estimator
- Model
- Pipeline
- PipelineModel

pyspark.ml.Transformer

Abstract class for transformers that transform one dataset into another.

pyspark.ml.Estimator

Abstract class for estimators that fit models to data.

pyspark.ml.Model

Abstract class for models that are fitted by estimators.

More Plainer English

An Estimator produces a Model for a given DataFrame and parameters (as ParamMap).

It fits a model to the input DataFrame and ParamMap to produce a Transformer (a Model) that can calculate predictions for any DataFrame-based input datasets.

It's basically a function that maps a DataFrame onto a Model through fit method, i.e. it takes a DataFrame and produces a Transformer as a Model.

```
estimator: DataFrame =[fit]=> Model
```

pyspark.ml.Pipeline

A simple pipeline, which acts as an estimator.

A Pipeline consists of a sequence of stages, each of which is either an Estimator or a Transformer.

When Pipeline.fit() is called, the stages are executed in order.

If a stage is an Estimator, its Estimator.fit() method will be called on the input dataset to fit a model. Then the model, which is a transformer, will be used to transform the dataset as the input to the next stage.

If a stage is a Transformer, itsTransformer.transform() method will be called to produce the dataset for the next stage.

pyspark.ml.Pipeline

The fitted model from a Pipeline is a PipelineModel, which consists of fitted models and transformers, corresponding to the pipeline stages.

If stages is an empty list, the pipeline acts as an identity transformer.

pyspark.ml.PipelineModel

Represents a compiled pipeline with transformers and fitted models.

Serialize Your Models

Serialization

The process of translating data structures or object state into a format that can be stored (for example, in a file or memory buffer) or transmitted (for example, across a network connection link) and reconstructed later (possibly in a different computer environment).

How do you serialize a linear model?

$$\hat{Y} = \varepsilon+\beta_0 x_0+\beta_1 x_1+\ldots+\beta_n x_n$$

Serialization: Sklearn

```python
import pickle

# Dump a Scikit-Learn fitted estimator to disk
with open(path, 'wb') as f:
    pickle.dump(model, f)

# Load a Scikit-Learn fitted estimator from disk
with open(path, 'rb') as f:
    model = pickle.load(f)
```

Serialization: Spark MLlib

```
from pyspark.ml.regression import LinearRegressionModel

# Dump a PySpark fitted model to disk
lr_model.write().overwrite().save(MODEL_FILE_PATH)

# Load a PySpark fitted model to disk
new_lr_model =
LinearRegressionModel.load(MODEL_FILE_PATH)
```

A Tour of Model Families

Models: Instance Methods

Compare instances in data set with a similarity measure to find best matches.

- Suffers from curse of dimensionality.
- Focus on feature representation and similarity metrics between instances

k-Nearest Neighbors (kNN)
Self-Organizing Maps (SOM)
Learning Vector Quantization (LVQ)

Models: Regression

Model relationship of independent variables, X to dependent variable Y by iteratively optimizing error made in predictions.

Ordinary Least Squares
Logistic Regression
Stepwise Regression
Multivariate Adaptive Regression Splines (MARS)
Locally Estimated Scatterplot Smoothing (LOESS)

Models: Regularization Methods

Extend another method (usually regression), penalizing complexity (minimize overfit)

- simple, popular, powerful
- better at generalization

Ridge Regression
LASSO (Least Absolute Shrinkage & Selection Operator)
Elastic Net

Models: Decision Trees

Model of decisions based on data attributes. Predictions are made by following forks in a tree structure until a decision is made. Used for classification & regression.

Classification and Regression Tree (CART)
Decision Stump
Random Forest
Multivariate Adaptive Regression Splines (MARS)
Gradient Boosting Machines (GBM)

Models: Bayesian

Explicitly apply Bayes’ Theorem for classification and regression tasks. Usually by fitting a probability function constructed via the chain rule and a naive simplification of Bayes.

Naive Bayes
Averaged One-Dependence Estimators (AODE)
Bayesian Belief Network (BBN)

Models: Kernel Methods

Map input data into higher dimensional vector space where the problem is easier to model. Named after the “kernel trick” which computes the inner product of images of pairs of data.

Support Vector Machines (SVM)
Radial Basis Function (RBF)
Linear Discriminant Analysis (LDA)

Models: Clustering Methods

Organize data into into groups whose members share maximum similarity (defined usually by a distance metric). Two main approaches: centroids and hierarchical clustering.

k-Means
Affinity Propagation
OPTICS (Ordering Points to Identify Cluster Structure)
Agglomerative Clustering

Models: Artificial Neural Networks

Inspired by biological neural networks, ANNs are nonlinear function approximators that estimate functions with a large number of inputs.

- System of interconnected neurons that activate
- Deep learning extends simple networks recursively

Perceptron
Back-Propagation
Hopfield Network
Restricted Boltzmann Machine (RBM)
Deep Belief Networks (DBN)

Models: Ensembles

Models composed of multiple weak models that are trained independently and whose outputs are combined to make an overall prediction.

Boosting
Bootstrapped Aggregation (Bagging)
AdaBoost
Stacked Generalization (blending)
Gradient Boosting Machines (GBM)
Random Forest

Models: Other

The list before was not comprehensive, other algorithm and model classes include:

Conditional Random Fields (CRF)
Markovian Models (HMMs)
Dimensionality Reduction (PCA, PLS)
Rule Learning (Apriori, Brill)
More ...

Spark MLlib Algorithms

Spark MLlib: Classification

Binary Classification

- Linear SVMs
- Logistic regression
- Decision trees
- Random forests
- Gradient-boosted trees
- Naive Bayes

Multiclass Classification

- Logistic regression
- Decision trees
- Random forests
- Naive Bayes

Spark MLlib: Regression

- Linear least squares
- Lasso
- Ridge regression
- Decision trees
- Random forests
- Gradient-boosted trees
- Isotonic regression

Spark MLlib: Clustering

- K-means
- Gaussian mixture
- Power iteration clustering (PIC)
- Latent Dirichlet Allocation (LDA)
- Bisecting k-means
- Streaming k-means

Spark MLlib: Dimensionality Reduction

- Singular Value Decomposition (SVD)
- Principal component analysis (PCA)

Spark MLlib: Feature Extraction

- TF-IDF (HashingTF, IDF)
- Word2Vec
- StandardScalar
- Normalizer
- ChiSqSelector
- ElementwiseProduct
- PCA

Spark MLlib: Evaluation Metrics

- Classification
  - Binary
    - Precision
    - Recall
    - F-measure
    - ROC
    - Area Under ROC
    - Area Under Precision-Recall Curve

Spark MLlib: Evaluation Metrics

- Classification
    - Multiclass
        - Confusion matrix
        - Accuracy
        - Precision by label
        - Recall by label
        - F-measure by label
        - Weighted precision
        - Weighted Recall
        - Weighted F-measure

Spark MLlib: Evaluation Metrics

- Classification
  - Multilabel
    - Precision
    - Recall
    - Accuracy
    - Precision by label
    - Recall by label
    - F-measure by label
    - Weighted precision
    - Weighted Recall
    - Weighted F-measure

Spark MLlib: Evaluation Metrics

- Regression
  - Mean Squared Error (MSE)
  - Root Mean Squared Error (RMSE)
  - Mean Absolute Error (MAE)
  - Coefficient of Determination (R^2)
  - Explained Variance

Hands-On Lab

Tasks

Sampling

- Pandas
- PySpark

Serialization

- Sklearn
- PySpark

Modeling in PySpark

- Classification: Naive Bayes
- Regression: Linear Regression