Machine Learning & MapReduce

MLbase: A Distributed Machine Learning System
Map-Reduce for Machine Learning on Multicore

Minjae Kim, Leonhard Spiegelberg
04/15/2016

1. Machine Learning

1.1 What is the problem?

→ Overwhelming zoo and complexity of ML algorithms / ML training

→ ML system separate from database system

→ Existing systems require background in distributed systems and algorithms

Regression Decision Trees SVMs Naive Bayes kNN PCA Clustering

What are the key challenges
for modern machine learning?

1.2 Key Challenges

→ scalability

→ parametrization and model selection

→ training

→ real-time / interactive responses

What differentiates a ML system
from a traditional database system?

1.3 System design

→ Sampling as important operation

→ No need for complicated index structures

- → Joins, filtering and aggregation usually done before training
- → Raw data not updated during training / prediction

→ **Scans & loops for training** (stochastic gradient descent [SGD])

1.3 Stochastic Gradient Descent (SGD)

Objective function

$$Q(\theta) = \sum_{i=1}^{n} Q_i(\theta)$$

Gradient descent

$$\theta^{(k+1)} = \theta^{(k)} - \eta \sum_{i=1}^{n} \nabla Q_i(\theta)$$

Approximate gradient

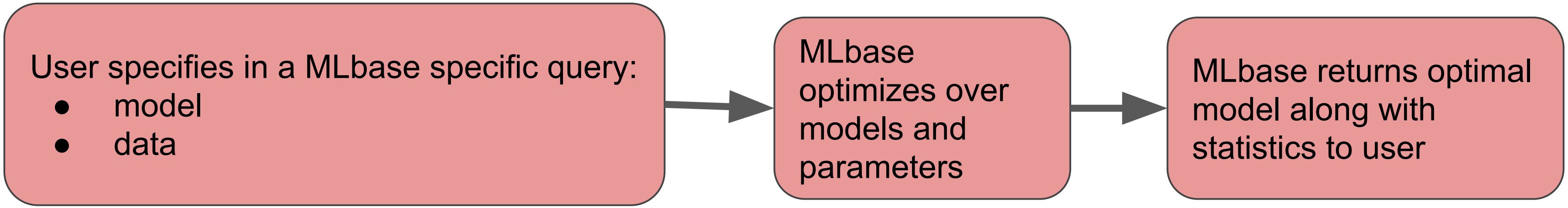
$$\sum_{i=1}^{n} \nabla Q_i(\theta) \approx \frac{n}{|J|} \sum_{j \in J} \nabla Q_j(\theta)$$

(Standard) Algorithm:

- Choose an initial vector of parameters w and learning rate η.
- Repeat until an approximate minimum is obtained:
  - Randomly shuffle examples in the training set.
  - For $i = 1, 2, ..., n$, do:
    - $w := w - \eta \nabla Q_i(w)$.

1.4 The solution

MLbase:



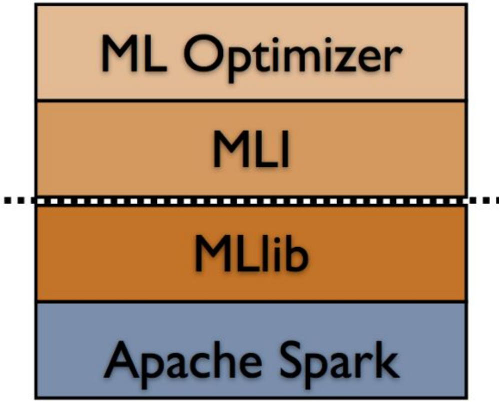

```
var X = load("als_clinical", 2 to 10)
var y = load("als_clinical", 1)
var (fn-model, summary) = doClassify(X, y)
```

1.5 Core solution

MLbase optimizes over models and parameters

Create logical learning plan (**LLP**) → Create and execute physical learning plan (**PLP**)

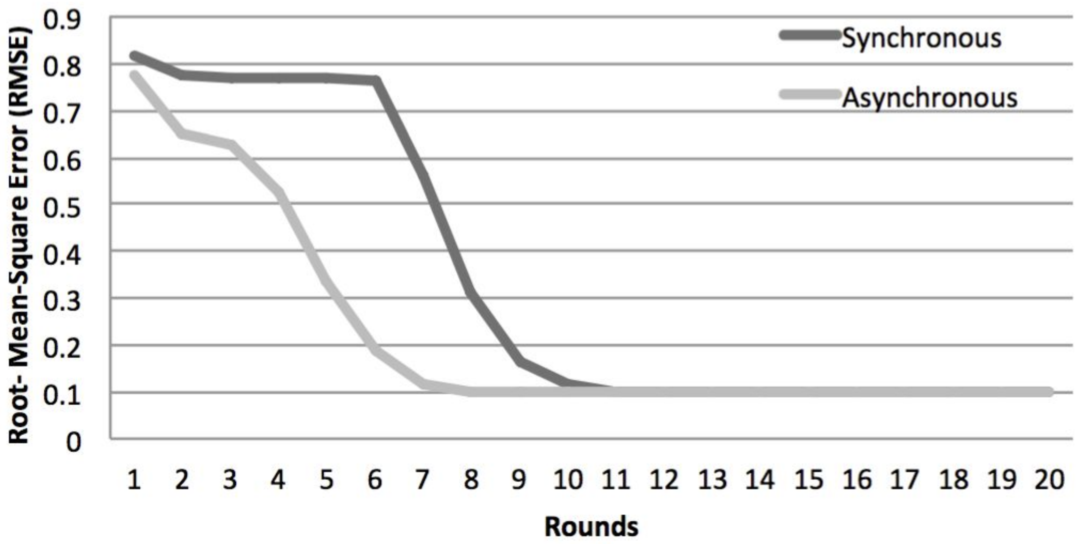
MLbase Runtime



1.6 Step by Step solution ideas of MLbase

- Do optimization only on 10% of data
- Each model can be trained separately
- Relax certain db assumptions
- Give user early results and continuously improve

1.7 Synchronous vs. asynchronous execution



→ Optimization routines do not require consistency

What design choices could we make
to create a more ML friendly database system?


Learning
Kernel
Computer
Vector
Supervised
Linear
Neural
Regression
Parallelization
Stochastic
DAG
Unsupervised
Clustering
Prior
Hidden
MAP
Stochastic
Models
Posterior
Model
Carlo
Grid
Bias
Learning
MCMC
Perceptron
Maximization
models
Learning
Set
Models
Gibbs
Bayesian
Cross-Validation
Train
functions
Graphical
Variance
Models
Better
Reduction
Easy
LDA
Deep
Machines
MLE
Set
kNN
Complex
Mixture
Inference
Descenta
kMeans
Test
Statistics
Models
Latent
Markov
Expectation
Search
Monte
Regression
Gradient
Vision
Variables
Networks
Logistic
Sampling
Support
Processes
Basis

1.8 Ideas & next steps regarding MLbase

→ Implement system & design appropriate experiments

→ exchange Spark with other frameworks

→ add feature extraction to optimization routine

→ use globally collected statistics for parameter search

2. MapReduce

2.1 Statistical Query Model

ML requires computing statistical quantities (e.g. moments)

$$\sum_{x \in D} f(x) = f(x_1) + f(x_2) + \ldots$$

How to parallelize?

2.2 MapReduce = Map + Reduce?

$$\sum_{x \in D} f(x) = \underbrace{\boxed{f(x_1)}}_{\text{Core 1}} + \underbrace{\boxed{f(x_2)}}_{\text{Core 2}} + \ldots$$

2.2 MapReduce = Map + Reduce?

$$\sum_{x \in D} f(x) = \boxed{f(x_1) + f(x_2)} + \ldots$$

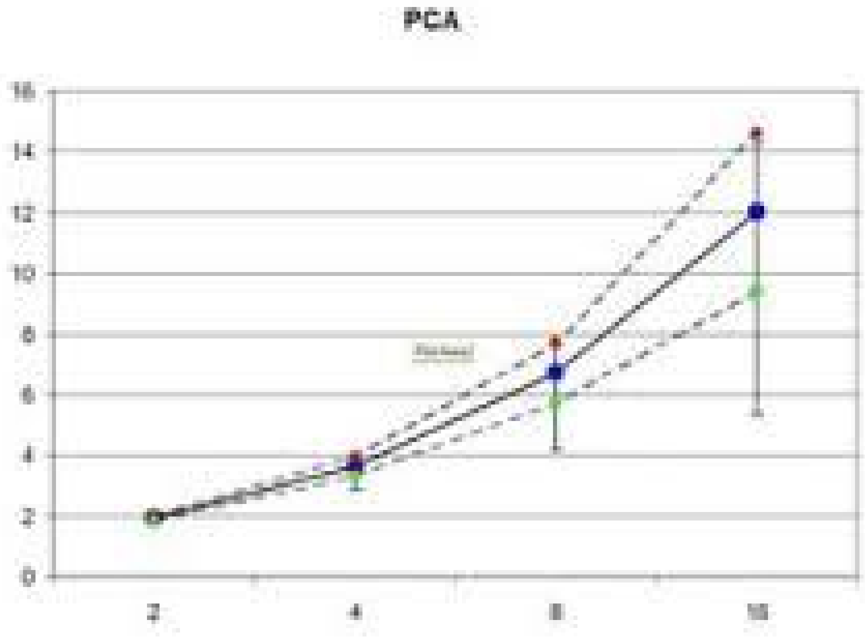
Core 1

Core 2...n

2.3 Result

Linear speedup (close to ideal)

e.g. PCA



Parallel programming solved?