MADlib and MLbase

Quan Fang
Xiaofeng Tao
MADlib Analytics Library
What is a mad lib?

Mad Libs

Data Scientist Job Description

• Define _____________________ collection requirements to enhance _____________________
  (synonym for data) (sciencey noun)

• Extract patterns from ___________________ and transform __________________________ into __________
  (scientific buzzword) (noun few people have heard of) (sexy noun)

• Perform automated ___________________ and entity _____________________________
  (scientific buzzword) (scientific buzzword)

- From bigdatablog.emc.com
What is a mad lib?

- From bigdatablog.emc.com
MAD Methodology


• What does MAD stand for?
Magnetic

• Use all data sources regardless of format or quality
Magnetic Agile

- Use all data sources regardless of format or quality
- Support dynamic workflow and whatever data analysts need
Magnetic Agile Deep

- Use all data sources regardless of format or quality
- Support dynamic workflow and whatever data analysts need
- Analyze large datasets without sampling
Magnetic Agile Deep

• Goal: Give analysts access to familiar math concepts, statistical methods, and algorithms for database

• Use traditional SQL and a wide range of extension languages
Data Parallel Statistics

• Give analysts access to familiar math concepts and statistical methods for database
• Database methods are data parallel and have an SQL like interface
• Database methods fall into one of the following layers of abstraction:
  – Scalar
  – Vector
  – Matrix
  – Function
MAD DBMS

• Loading and Unloading
  – Scatter / Gather streaming for fully parallel access
  – External tables

• Extract-Transform-Load (ETL) and Extract-Load-Transform (ELT)
MAD DBMS

• Multiple storage formats
  – External tables
  – Heap format
  – Append-mostly format
• Different storage formats per table partition
• Atomic partition exchange
MAD Programming

• Traditional SQL queries with extensions
• Map/Reduce functions written in Python, Perl, or R
• Wide range of extension languages
MAD Implementations

• MADlib
  – Library for data analytics that can run on Greenplum or PostgreSQL

• Greenplum database
  – Massively parallel processing database
MADlib

• Library for widely used statistics, data mining, and machine learning that runs on top of a SQL database
  – Regression models
  – Machine learning
  – Linear systems
  – Topic modelling
  – Descriptive statistics
  – And more…. (see doc.madlib.net)
MADlib History

• Still in early stages of development
• In use by some research universities and companies

• Heavily sponsored by Greenplum
MADlib Interface

- Interface consists of extensible SQL like scripts that call MADlib functions
- Wide range of extension languages possible but C++ / Python recommended
- Designed for portability
MADlib User Extensions

• User defined aggregates and functions in C++
• Driver functions use Python to wrap multiple MADlib SQL calls
• Templated queries are supported with Python wrappers
MADlib: Logistic regression

CREATE TEMPORARY TABLE iterative_algorithm AS
SELECT 0 AS iteration, NULL AS state

current_iteration = 0

INSERT INTO iterative_algorithm(iteration, state)
SELECT iteration + 1, logregr_irls_step(y, x, state)
FROM source, iterative_algorithm
WHERE iteration = current_iteration

current_iteration ++

SELECT internal_logregr_irls_did_converge(state)
FROM iterative_algorithm
WHERE iteration = currentIteration

[False]  [True]

SELECT internal_logregr_irls_result(state)
FROM iterative_algorithm
WHERE iteration = currentIteration
MADlib Example: K-means

• Run K-means

```sql
SELECT * FROM madlib.kmeanspp('km_sample', 'points', 2, 'madlib.squared_dist_norm2', 'madlib.avg', 20, 0.001 );
```
MADlib Example: K-means

- Calculate silhouette coefficient

```sql
SELECT * FROM madlib.simple_silhouette('km_sample', 'points',
(SELECT centroids FROM 'km_centroids'),
'madlib.squared_dist_norm2');
```
Live demo

• Postgres command line
• Principal Component Analysis
  – pca_train()
  – pca_project()
Conclusion

• Due to the economics of data (cheap, high volume), data warehouses in the future need to focus on data analytics
  – Magnetically attract all data sources
  – Agile analysis workflow
  – Deeply analyze large datasets without sampling

• MADlib provides analysts with such a toolbox that is portable across SQL databases
MLbase: A Distributed Machine-Learning System
Four foci

• (1) A new declarative language to specify ML tasks
• (2) A novel optimizer to select ML algorithms
• (3) Provide answers early and continuously refine.
• (4) Design a distributed run-time optimized for the data access patterns of ML
A Declarative Approach to ML
Spark: cluster computing system designed for iterative computation

MLlib: low-level ML library in Spark

MLI: API / platform for feature extraction and algorithm development
  • Platform independent

ML Optimizer: automates model selection
  • Solves a search problem over feature extractors and algorithms in MLI
MLbase Architecture

1. MLI: Interface between MLBase and Distributed Infra
2. Algorithms, Data, Statistics
3. Adaptive Optimizer

User

Declarative ML Task (e.g., fn-model & summary)

Master Server

- Meta-Data
- Binders of Algorithms
- Statistics

Parser

LLP

PLP

COML (Optimizer)

Executor/Monitoring

Runtime

ML Contract + Code

ML Developer

Result (e.g., fn-model & summary)
1 MLI: Machine Learning Interface

MLI Interface between MLBase and Distributed Infra
1 MLI: Machine Learning Interface

• Shield ML Developers from low-level-details.
• Independence between ML algorithm and run-time
• Current supported run-times:

   Spark

   TupleWare
Algorithms Pool

User

Declarative ML Task (e.g., fn-model & summary)

ML Contract + Code

Master Server

Parser

COML (Optimizer)

Executor/Monitoring

ML Developer

ML Code

Meta-Data

Binders of Algorithms

Statistics

Algorithms, Data, Statistics

Runnable

Result

Master
Algorithms Pool

Implementation
Algorithms pool

ML Base defined Contracts
- Type (e.g., classification)
- Parameters
- Runtime (e.g., O(n))
- Input-Specification
- Output-Specification
- ...

Extensibility
MLbase Architecture

User

Declarative ML Task
(e.g., fn-model & summary)

Master Server

Meta-Data

Binders of Algorithms

Statistics

Parser

COML (Optimizer)

LLP

PLP

Executor/Monitoring

ML Contract + Code

ML Developer

Optimizer LLP & PLP

Runtime

Runtime

Runtime

Runtime

Master
Query Optimization (Logical Learning Plan)

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

(2) Generic Logical Plan
load (als_clinical)
(X, y)
down-sample
(X', y')
grid-search
configure model

(3) Optimized Plan
load (als_clinical)
(X, y)
down-sample 10%
(X', y')
standard feature normalizer
(X'', y'')
create 10-folds
store normalized folds
folds
cross validation
SVM
kernel: RBF
\lambda = 10^6, \sigma = 1/d \times 10^6
cross validation
SVM
kernel: RBF
\lambda = 10^6, \sigma = 1/d \times 10^6
cross validation
AdaBoost
rounds = 20
top-1
train model
baseline-check: most common label
calculate misclassification rate
top-1
fn-model
(train-model, cross-validation-summary)

Figure 2: Optimization Process
Run Time (Physical Learning Plan)

• (1) The master distributes ML tasks to workers
• (2) Monitor progress
• (3) Handle failures
Conclusion

• MLbase’s core is its optimizer
• Transforms a declarative ML task to a sophisticated learning plan
• Quickly returns a first quality answer, improves in the background
• To be fully distributed and offers a run-time able to exploit ML algorithms