Introduction to Data Science with Hadoop

Glynn Durham | Senior Instructor
glynn@cloudera.com
May 2017
Short and Sweet
Hadoop
What About Spark?
Machine Learning
The Future
Short and Sweet
Hadoop
What About Spark?
Machine Learning
The Future
Data Science is ...

- gathering data, potentially of many types and from many sources,
- wrangling that data into useful forms, and
- applying statistical programming and machine learning, to gain new information from the data.
Machine Learning and Data Volume

“It’s not who has the best algorithms who wins. It’s who has the most data.”

[Banko and Brill, 2001]
Hadoop is ...

• an open source software platform for
• acquiring, storing, and processing massive volumes of data,
• economically.
The Age of Machine Learning
Short and Sweet
Hadoop
What About Spark?
Machine Learning
The Future
The word “Hadoop” means

• a child’s toy
  or
• Hadoop Core
  or
• the Hadoop Ecosystem.
Hadoop Core

• A free open source software software project
• Managed transparently online, at the Apache Software Foundation (ASF), apache.org
• The project was started in 2006, based on papers from Google, in 2003 and 2004

• Consists of:
  • HDFS (Hadoop Distributed File System), for storage
  • Hadoop MapReduce, for processing
  • YARN (Yet Another Resource Negotiator)
Hadoop Core main features:
File storage and batch programming

*Save* my file with name

/user/joe/logfiles/20130429.tsv

*Read* file(s)

/user/alice/emails/*

*Run my program* "summarize.jar",
reading input

/user/eddie/national_sales/us/*
and writing output into new directory

/user/eddie/us_sales_summary/
HDFS Writes

Write file /user/geo/log.txt
HDFS Reads

Read file /user/geo/log.txt

Read part of /user/geo/log.txt
General File Input/Output

There are a variety of ways to copy any file into HDFS, included with Hadoop Core:

- Single line command:
  ```bash
  hadoop fs
  -put myfile hadfile
  ```

- FUSE
  ```bash
  cp myfile /hdfs/hadfile
  ```

- Java program using HDFS API

- HttpFS
  ```bash
  curl "http://host:14...
  ```

Each of these has a complement for copying a file from HDFS
HDFS Strengths and Weaknesses

• HDFS is good at:
  • storing enormous files
  • storing lots of data reliably
  • throughput on sequential writes
  • throughput on sequential reads of a file or part of a file

• HDFS is not good at:
  • high speed (low latency) random reads of parts of a file

• HDFS cannot:
  • update any part of a file once written*

* but you can always write a new file and/or delete, move, and rename files and directories
MapReduce: Programming with simple functions

Input records

Map function takes one input record, returns 0 or more intermediate records

Intermediate records must be of the form (key, value)

Shuffle sorts records by key

Reduce function takes records of one key, returns 0 or more output records

Output records
MapReduce Example: Word Count

Count the number of occurrences of each word over a large amount of input data

• This is the ‘hello world’ of MapReduce programming

```java
map(String input_key, String input_value)
    foreach word w in input_value:
        emit(w, 1)
```

```java
reduce(String output_key,
        Iterator<int> intermediate_vals)
    set count = 0
    foreach v in intermediate_vals:
        count += v
    emit(output_key, count)
```
Word Count, continued

Input to the Mapper:

(3414, 'the cat sat on the mat')
(3437, 'the aardvark sat on the sofa')

Output from the Mapper:

('the', 1), ('cat', 1), ('sat', 1), ('on', 1),
('the', 1), ('mat', 1), ('the', 1), ('aardvark', 1),
('sat', 1), ('on', 1), ('the', 1), ('sofa', 1)
Word Count, continued

Intermediate data sent to the Reducer:

(('aardvark', [1])
('cat', [1])
('mat', [1])
('on', [1, 1])
('sat', [1, 1])
('sofa', [1])
('the', [1, 1, 1, 1]))

Final Reducer output:

(('aardvark', 1)
('cat', 1)
('mat', 1)
('on', 2)
('sat', 2)
('sofa', 1)
('the', 4))
So we just counted words. So what?

• Many problems conform to this pattern:
  • **Web log analysis**: map() emits an IP address for each web log event; reduce() counts occurrences for each IP address
  • **Indexing**: For each document, map() emits each term of interest paired with the document ID; reduce() collects and emits all document IDs for each term
  • **Page rank algorithm**:
    • Every web page (URL) on the Web gets an initial score.
    • map() divides a page’s score among all of its outlinks’ URLs; reduce() sums the received scores for each URL.
    • Iterate on this procedure.
MapReduce Chains
MapReduce at Scale
MapReduce Strengths and Weaknesses

• MapReduce is good at:
  • processing enormous volumes of data
  • scaling out as you add more machines
  • continuing to completion, even when some machines die

• MapReduce is not good at:
  • running any algorithm you can write in pseudocode
  • algorithms that require shared state overall*
    * but maybe you can get clever with your algorithm design

• MapReduce cannot:
  • run in real time: MapReduce jobs are batch jobs
YARN, Yet Another Resource Negotiator

Run my MapReduce (MR V2)
or other type of program

Daemons spawn other processes that perform MapReduce processing, or—in time—programs in other paradigms.
Sqoop: RDBMS to Hadoop and Back

- Uses MapReduce to run concurrent database queries that extract or insert data
Flume: Ingesting Ongoing Event Data

Agent processes continually ingesting digital events into Hadoop
Kafka: General Data Streaming

kafka.apache.org
HBase: A NoSQL Database System

- A scalable key/value store
- Accommodates general binary data
- High volume, high performance access to individual items
- Random reads and writes
- Weaker query language than SQL (put, get, scan, delete)
- Lacks ACID-compliant transactions
Kudu: Scalable storage for structured data

kudu.apache.org
Hive: MapReduce (or Spark) as “SQL”

- Familiar language and programming paradigm
- Provides interface to many SQL-compliant tools

```sql
INSERT OVERWRITE TABLE 'summary'
SELECT region.name, SUM(order_total) region_sales
FROM region JOIN sales
ON (region.id = sales.region_id)
WHERE sales.sale_date > "20121231"
GROUP BY region.name
ORDER BY region_sales DESC;
```
Pig: Another Language for MapReduce (or Spark)

```
sales = LOAD 'sales' AS (orderId:INT, customerName:CHARARRAY,
  sale_date:INT, regionId:INT, orderTotal:FLOAT);
thisYearSales = FILTER sales BY sale_date > 20121231
region = LOAD 'region' AS (id:INT, name:CHARARRAY);
joined = JOIN region BY id, thisYearSales BY regionId;
grouped = GROUP joined BY region.id, region.name;
summary = FOREACH grouped GENERATE group.region.id,
  SUM(joined.orderTotal);
STORE summary INTO 'summary';
```
Impala: High Speed Analytics in Hadoop

- Purpose-built for high speed analytic queries
- Does not use MapReduce or Spark
- Usually 5 to 30 times faster—sometimes 100 times faster!

incubator.apache.org/projects/impala.html

```
SELECT region.name, SUM(order_total) region_sales
FROM region JOIN sales
ON (region.id = sales.region_id)
WHERE sales.sale_date > '20121231'
GROUP BY region.name
ORDER BY region_sales DESC;
```
And More

• Serialization and efficient file storage: Avro and Parquet

  avro.apache.org  parquet.apache.org

• Workflow: Oozie

  oozie.apache.org
And Even More...

- Security: Sentry and Record Service
  - [Sentry](sentry.apache.org)
  - [Record Service](recordservice.io)

- Machine Learning in MapReduce: Mahout
  - [Mahout](mahout.apache.org)

- And ...
Short and Sweet
Hadoop
What About Spark?
Machine Learning
The Future
Spark: An Improvement on MapReduce

• Originally a research project at UC Berkeley RAD Lab—later the AMPLab, in 2009

• Addresses some fundamental pain points of MapReduce

• The Spark Streaming subproject of 2012 adds near real-time programming
  • using “micro-batches” as an adaptation of batch programming
  • a capability altogether lacking in Hadoop MapReduce
Similarities of MapReduce and Spark

- Processes massive volumes of data with a scale-out, distributed framework
- The framework provides reliability, even in the face of machine failure
- Programming with stateless functions
- Relies on expensive shuffle to reorganize data for aggregation, joins, sorting
- Still lacks a shared state among all processes
- Can run under YARN to share processing resources
Improved API

- First-class APIs in Scala, Java, Python and R
- Data-flow programming paradigm (like Pig)
- Interactive shell—great for exploratory work
- Improved support for structured data and SQL-like processing
Processing Chains, Improved

Tasks, not new processes (JVMs)

Enhanced caching in memory

Eliminate I/O

Reduce I/O

Eliminate I/O
Spark MLlib: Machine Learning in Spark

- Subproject of Spark
- Effectively replaces Mahout for machine learning in Hadoop clusters
- From spark.apache.org, the front page:

But just be clear what you mean by “Hadoop”!

Logistic regression in Hadoop and Spark
Commercial Message # 1
Big Ecosystem

oozie.apache.org
Complete Big Data Platform

• Cloudera Manager can
  • install, monitor, manage, upgrade
    a coherent bundle of these projects and more

• Cloudera Director can
  • easily configure and deploy this platform on cloud services
    from Amazon, Google, or Microsoft

• !!!
Short and Sweet
Hadoop
What About Spark?
Machine Learning
The Future
Machine Learning Algorithms

• Supervised Learning:
  • Start with correctly labeled records, and learn to estimate or predict labels for new records
  • Continuous labels: Regression
  • Discrete labels: Logistic Regression, Classifiers

• Unsupervised Learning:
  • Start with unlabeled records, try to tease patterns (labels) out of the data
  • There is not a single “correct” answer for labeling
  • Continuous labels: Collaborative Filters (Recommenders)
  • Discrete labels: Clustering
Linear Regression: Supervised Learning of a Continuous Label

Max Acceptable Monthly Fee vs. Customer Income

- Max Acceptable Monthly Fee (USD)
- Annual Income (USD in Thousands)
Logistic Regression:
Supervised Learning of a Binary Label
Classifiers:
Supervised Learning of Discrete Labels

Training: Cat

Training: Table

Scoring: ???
Collaborative Filters (Recommenders): Unsupervised Learning of Continuous Labels

<table>
<thead>
<tr>
<th>Movie</th>
<th>Alice</th>
<th>Bob</th>
<th>Chuck</th>
<th>Donna</th>
<th>Eddie</th>
<th>Frank</th>
<th>Gina</th>
</tr>
</thead>
<tbody>
<tr>
<td>Airplane</td>
<td>1</td>
<td>4</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Bambi</td>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Caddyshack</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td></td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dracula</td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td>4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eat Pray Love</td>
<td>2</td>
<td>5</td>
<td>1</td>
<td></td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Friday</td>
<td>4</td>
<td></td>
<td></td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Gunsmoke</td>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Hang ‘Em High</td>
<td>5</td>
<td></td>
<td>4</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Iron Man</td>
<td></td>
<td></td>
<td>3</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>Jane Eyre</td>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>The Karate Kid</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Clustering: Unsupervised Learning of Discrete Labels
Spark MLlib: Machine Learning on Hadoop

Machine Learning Library (MLlib) Guide

MLlib is Spark's machine learning (ML) library. Its goal is to make practical machine learning scalable and easy. At a high level, it provides tools such as:

- ML Algorithms: common learning algorithms such as classification, regression, clustering, and collaborative filtering
- Featurization: feature extraction, transformation, dimensionality reduction, and selection
- Pipelines: tools for constructing, evaluating, and tuning ML Pipelines
- Persistence: saving and load algorithms, models, and Pipelines
- Utilities: linear algebra, statistics, data handling, etc.
Short and Sweet
Hadoop
What About Spark?
Machine Learning
The Future
Commercial Message # 2
More DS Teams in the Organization

- Collaboration, repeatability within teams
- Differing security requirements
- Different preferred programming languages: Python, R, Scala
- Different software libraries: Pandas, H2O, etc.
- Even different versions of software
Cloudera Data Science Workbench

- Development in Python, Scala, or R
- Differing security requirements
Deep Learning
Deep Learning on Hadoop

• Deep Learning refers to a category of classifier algorithms, mostly invented in 2006.
• Spark MLlib does not have any direct implementation of DL.
• There are several additional projects that can fit DL onto Spark/Hadoop:
  • BigDL
  • Caffe
  • TensorFlow
  • DL4J
The Road—or Runway(!)—Ahead

• It is a truism that organizations today have valuable insights hidden in their data that are waiting to be uncovered.
• 90% of all data that will exist in 2020 has yet to be created.
• Open source is here to stay.
• Hadoop as a data science platform is evolving, and its use is growing exponentially.
Thank you
glynn@cloudera.com