

INTELIGENCIA DE NEGOCIO

2019 - 2020



- **Tema 1. Introducción a la Inteligencia de Negocio**
- **Tema 2. Minería de Datos. Ciencia de Datos**
- **Tema 3. Modelos de Predicción: Clasificación, regresión y series temporales**
- **Tema 4. Preparación de Datos**
- **Tema 5. Modelos de Agrupamiento o Segmentación**
- **Tema 6. Modelos de Asociación**
- **Tema 7. Modelos Avanzados de Minería de Datos**
- **Tema 8. Big Data**

Big Data

Nuestro mundo gira en torno a los datos

■ Ciencia

- Bases de datos de astronomía, genómica, datos medio-ambientales, datos de transporte, ...



■ Ciencias Sociales y Humanidades

- Libros escaneados, documentos históricos, datos sociales, ...



■ Negocio y Comercio

- Ventas de corporaciones, transacciones de mercados, censos, tráfico de aerolíneas, ...



■ Entretenimiento y Ocio

- Imágenes en internet, películas, ficheros MP3, ...



■ Medicina

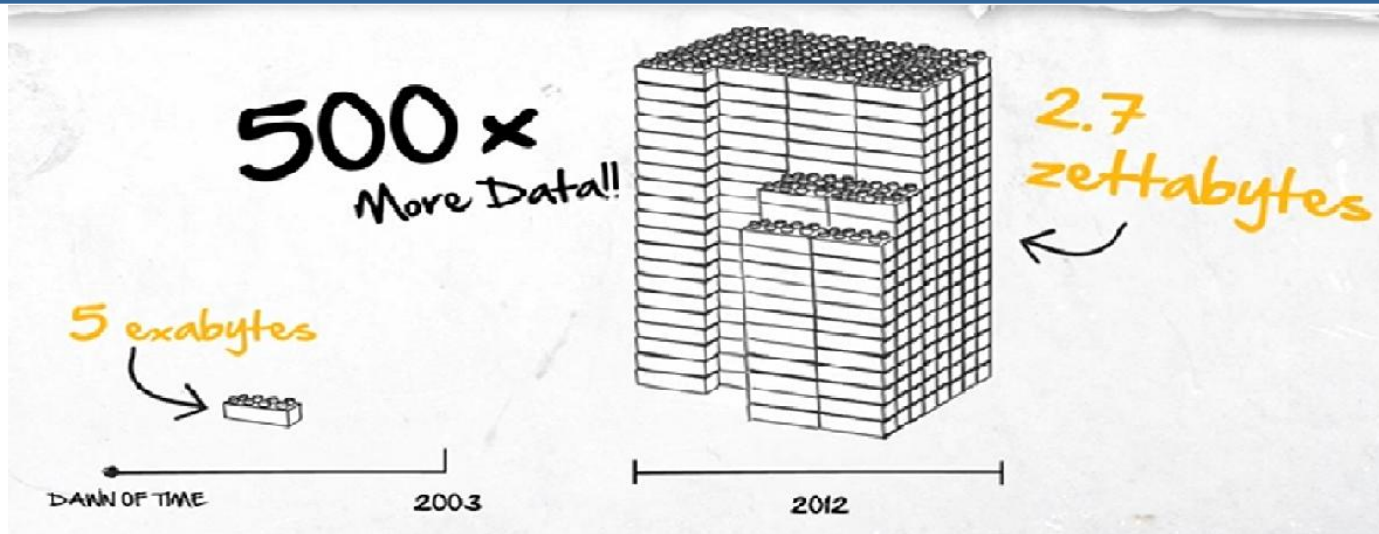
- Datos de pacientes, datos de escaner, radiografías ...

■ Industria, Energía, ...

- Sensores, ...

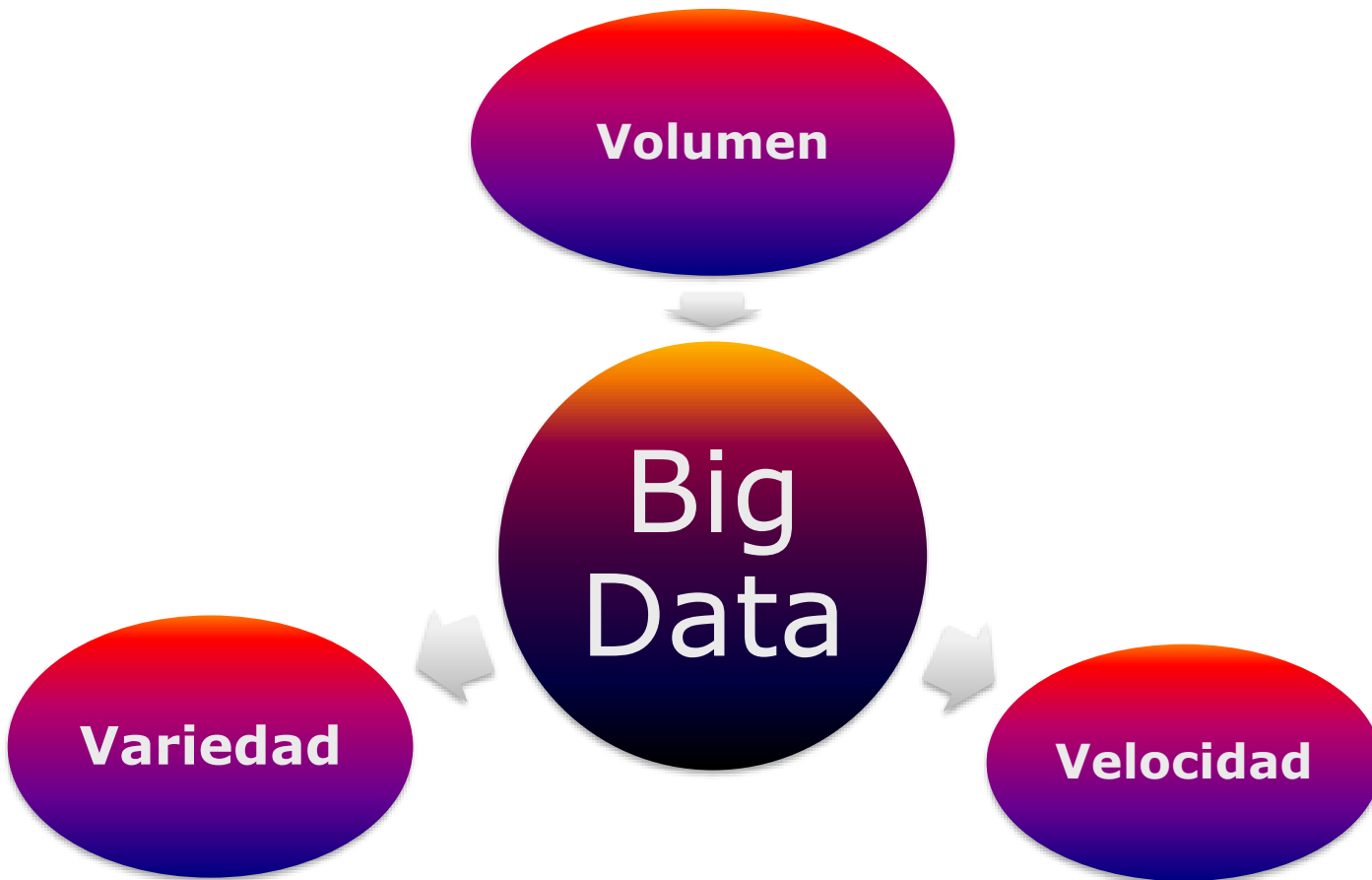


Big Data: La explosión de los datos



Big Data

Big Data en 3 V's



BIG DATA

Índice

- ❑ Big Data. Big Data Science
- ❑ ¿Por qué Big Data? Google crea el Modelo de Programación MapReduce
- ❑ Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...)
- ❑ Big Data Analytics: Librerías para Analítica de Datos en Big Data. Casos de estudio
- ❑ Algunas aplicaciones: Salud, Social Media, Identificación
- ❑ Comentarios Finales

BIG DATA

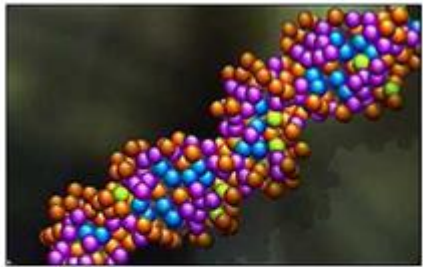
Índice

- ❑ **Big Data. Big Data Science**
- ❑ ¿Por qué Big Data? Google crea el Modelo de Programación MapReduce
- ❑ Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...)
- ❑ Big Data Analytics: Librerías para Analítica de Datos en Big Data. Casos de estudio
- ❑ Algunas aplicaciones: Salud, Social Media, Identificación
- ❑ Comentarios Finales

¿Qué es Big Data? 3 V's de Big Data



Ej. Genómica



- 25,000 genes in human genome
- 3 billion bases
- 3 Gigabytes of genetic data

Ej. Astronomía



- Astronomical sky surveys
- 120 Gigabytes/week
- 6.5 Terabytes/year

Ej. Transacciones de tarjetas de crédito



- 47.5 billion transactions in 2005 worldwide
- 115 Terabytes of data transmitted to VisaNet data processing center in 2004

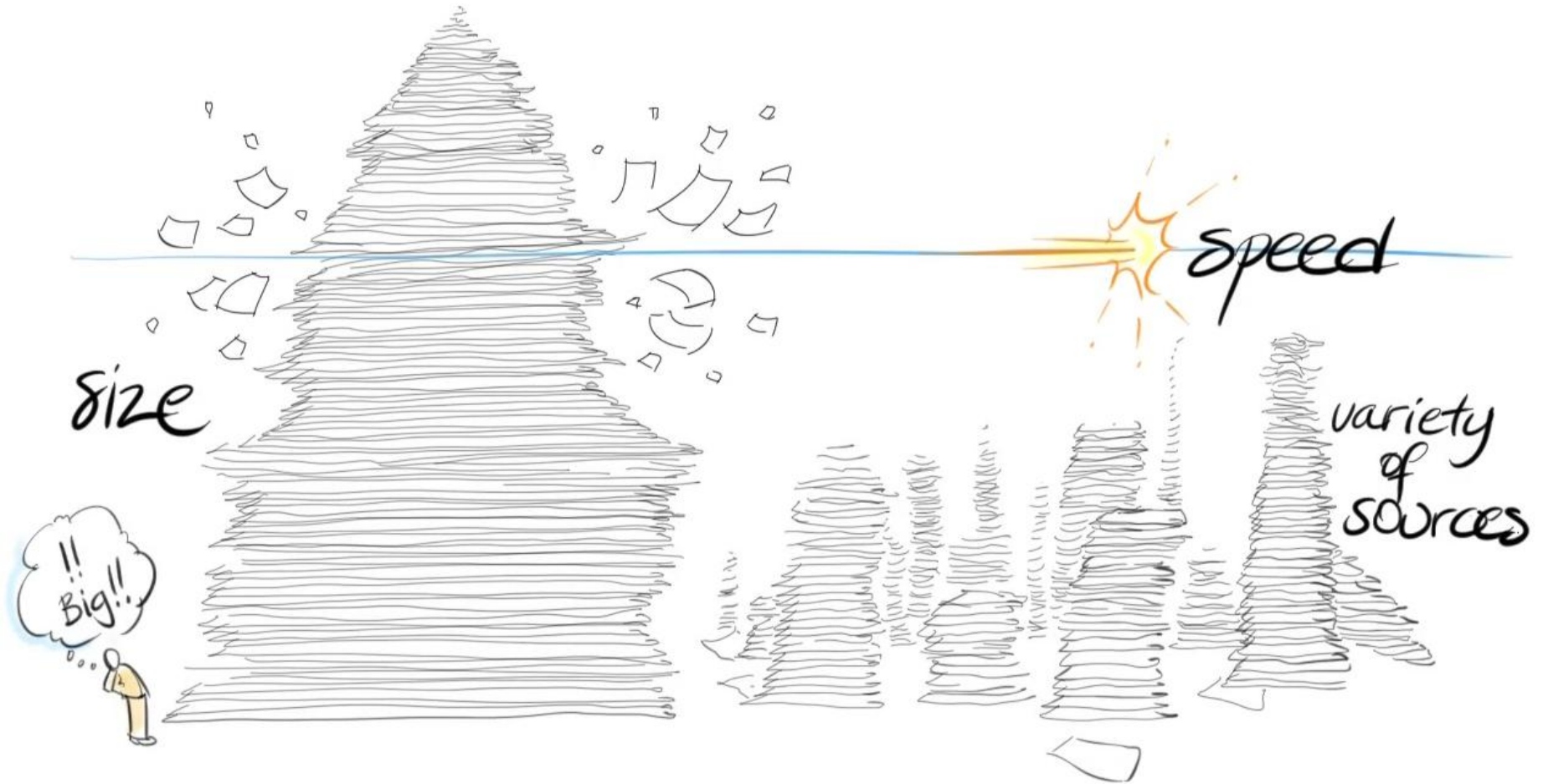
¿Qué es Big Data? 3 V's de Big Data



e PROMOTIONS

Ej. E-Promociones: Basadas en la posición actual e historial de compra → envío de promociones en el momento de comercios cercanos a la posición

¿Qué es Big Data? 3 V's de Big Data



¿Qué es Big Data? 3 V's de Big Data

Ej. Huella digital de pasajeros

Interior encarga un megacerebro capaz de localizar terroristas entre los pasajeros



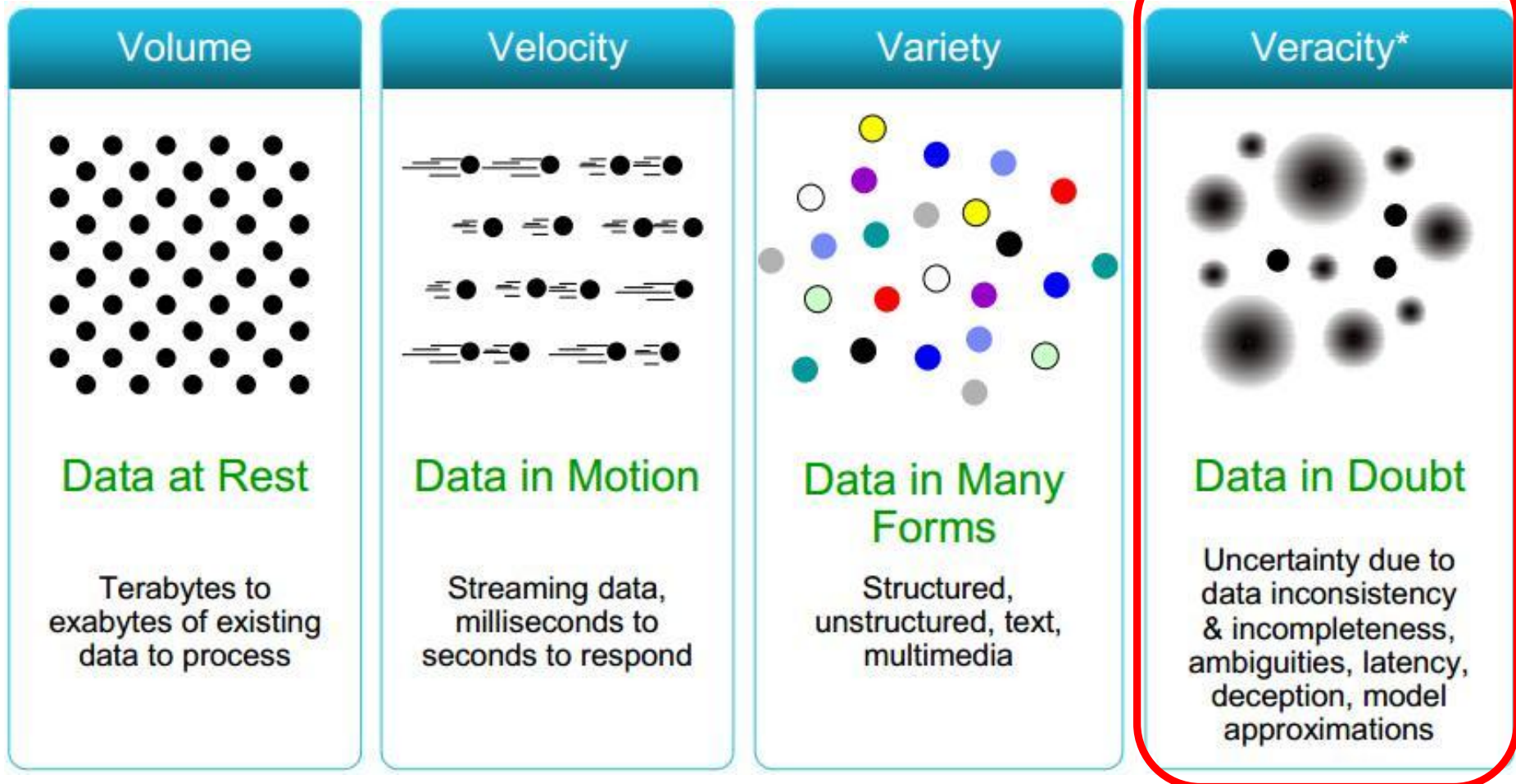
¿Un proyecto de ciencia ficción? El Ministerio del Interior cree que no. Que es posible tener un megacerebro electrónico capaz de localizar —a través de cálculos estadísticos y de cruzar ingentes cantidades de datos en milisegundos—...

"identificación automática del perfil demográfico y sociológico del pasajero"



¿Qué es Big Data? 3 V's de Big Data

Some Make it 4V's: Veracity



¿Qué es Big Data?

40 ZETTABYTES
[43 TRILLION GIGABYTES]
of data will be created by 2020, an increase of 300 times from 2005

6 BILLION PEOPLE
have cell phones

WORLD POPULATION: 7 BILLION

Volume
SCALE OF DATA

It's estimated that **2.5 QUINTILLION BYTES**
[2.3 TRILLION GIGABYTES]
of data are created each day

Most companies in the U.S. have at least **100 TERABYTES**
[100,000 GIGABYTES]
of data stored

The New York Stock Exchange captures **1 TB OF TRADE INFORMATION** during each trading session

Modern cars have close to **100 SENSORS** that monitor items such as fuel level and tire pressure

Velocity
ANALYSIS OF STREAMING DATA

By 2016, it is projected there will be **18.9 BILLION NETWORK CONNECTIONS**
— almost 2.5 connections per person on earth

The FOUR V's of Big Data

From traffic patterns and music downloads to web history and medical records, data is recorded, stored, and analyzed to enable the technology and services that the world relies on every day. But what exactly is big data, and how can these massive amounts of data be used?

As a leader in the sector, IBM data scientists break big data into four dimensions: **Volume, Velocity, Variety and Veracity**

Depending on the industry and organization, big data encompasses information from multiple internal and external sources such as transactions, social media, enterprise content, sensors and mobile devices. Companies can leverage data to adapt their products and services to better meet customer needs, optimize operations and infrastructure, and find new sources of revenue.

By 2015 **4.4 MILLION IT JOBS** will be created globally to support big data, with 1.9 million in the United States

As of 2011, the global size of data in healthcare was estimated to be **150 EXABYTES**
[161 BILLION GIGABYTES]

By 2014, it's anticipated there will be **420 MILLION WEARABLE, WIRELESS HEALTH MONITORS**

Variety
DIFFERENT FORMS OF DATA

4 BILLION+ HOURS OF VIDEO are watched on YouTube each month

30 BILLION PIECES OF CONTENT are shared on Facebook every month

400 MILLION TWEETS are sent per day by about 200 million monthly active users

1 IN 3 BUSINESS LEADERS don't trust the information they use to make decisions

Poor data quality costs the US economy around **\$3.1 TRILLION A YEAR**

Veracity
UNCERTAINTY OF DATA

27% OF RESPONDENTS in one survey were unsure of how much of their data was inaccurate

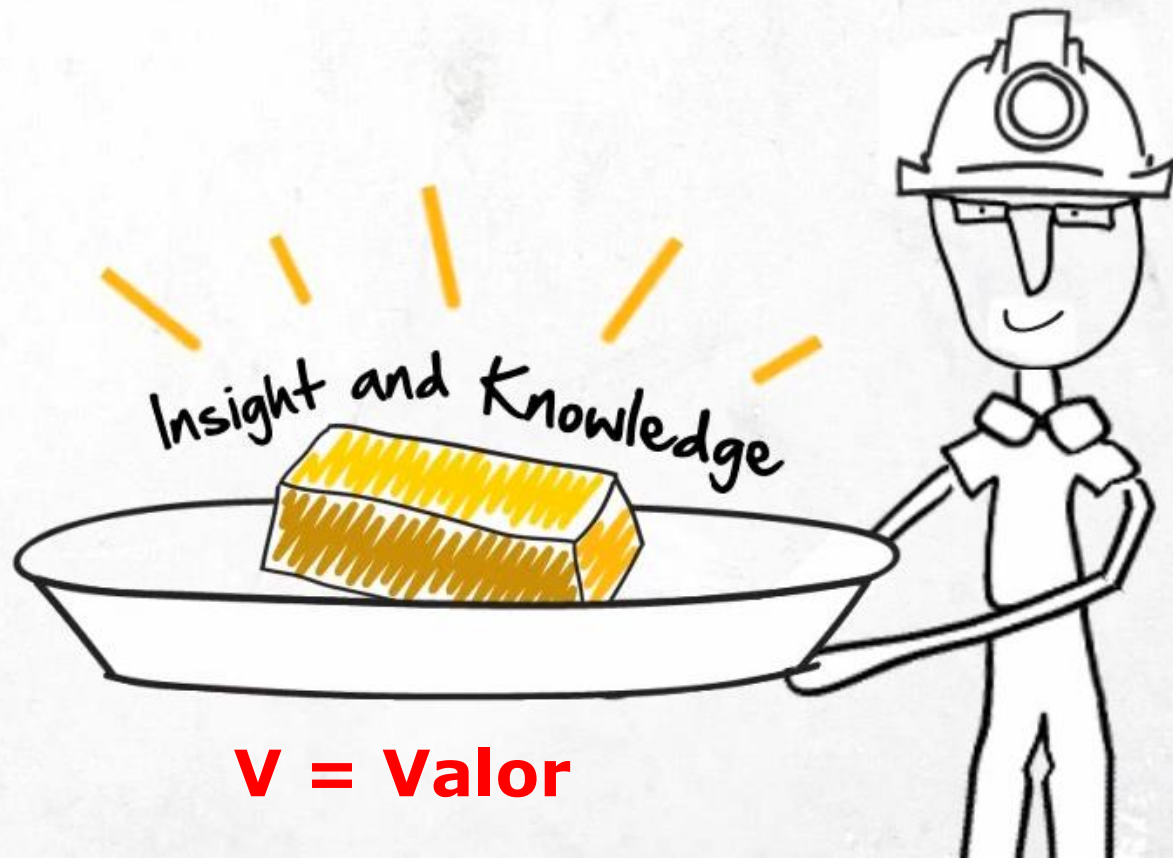
¿Qué es Big Data?

V --> Valor

Aproximaciones
y tecnologías
innovativas



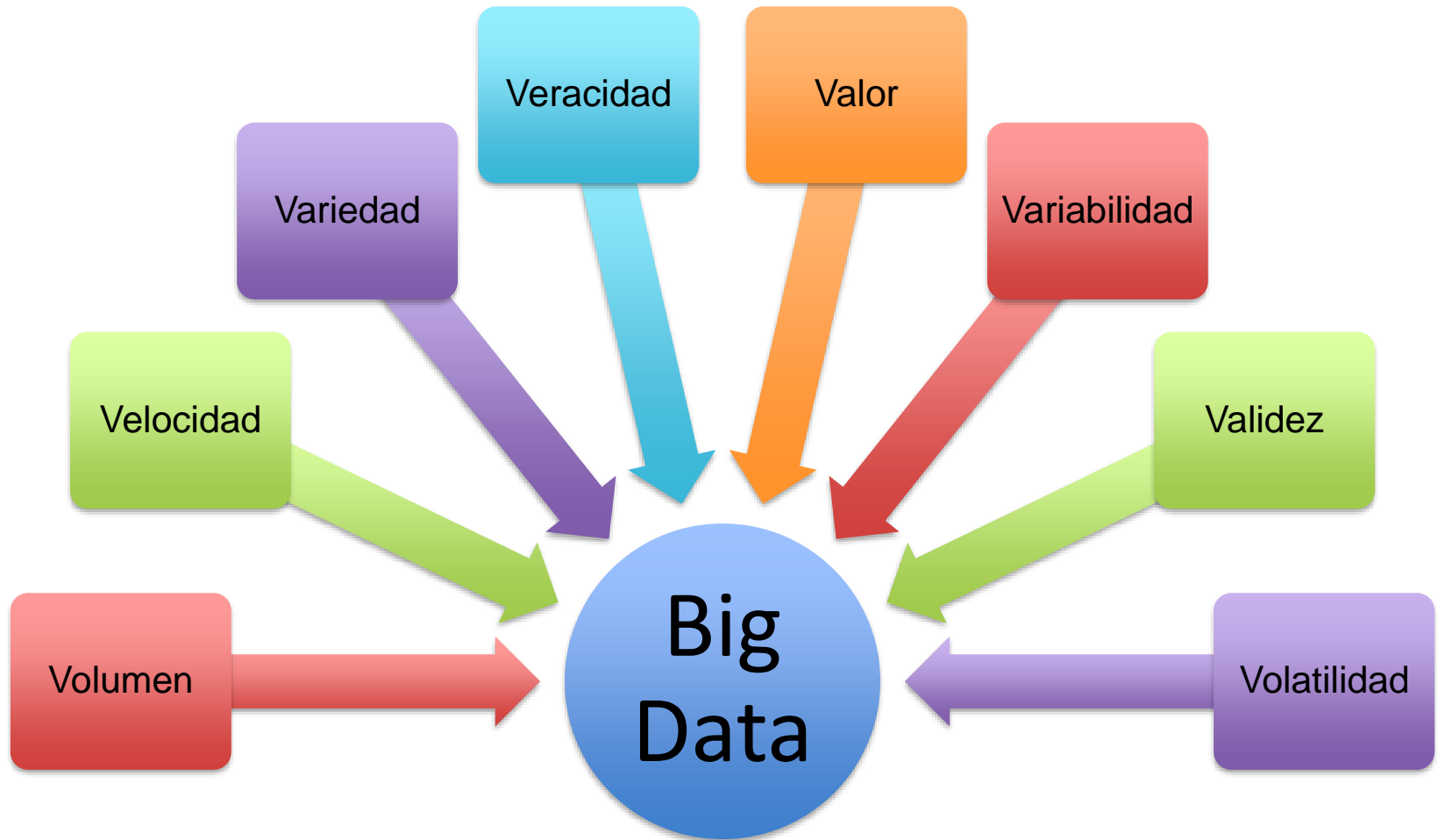
MapReduce



V = Valor

¿Qué es Big Data?

Las 8 V's de Big Data



¿Qué es Big Data?

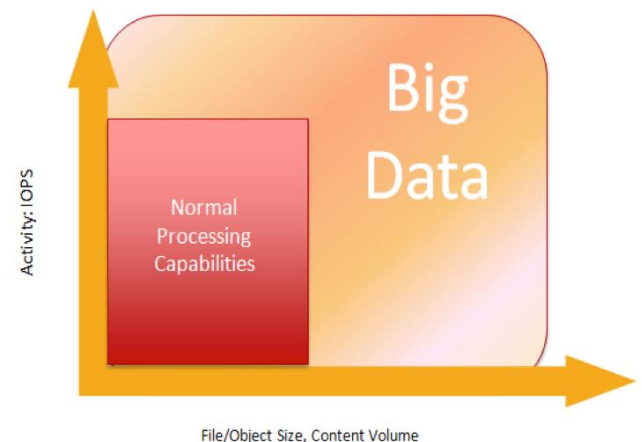
No hay una definición estándar



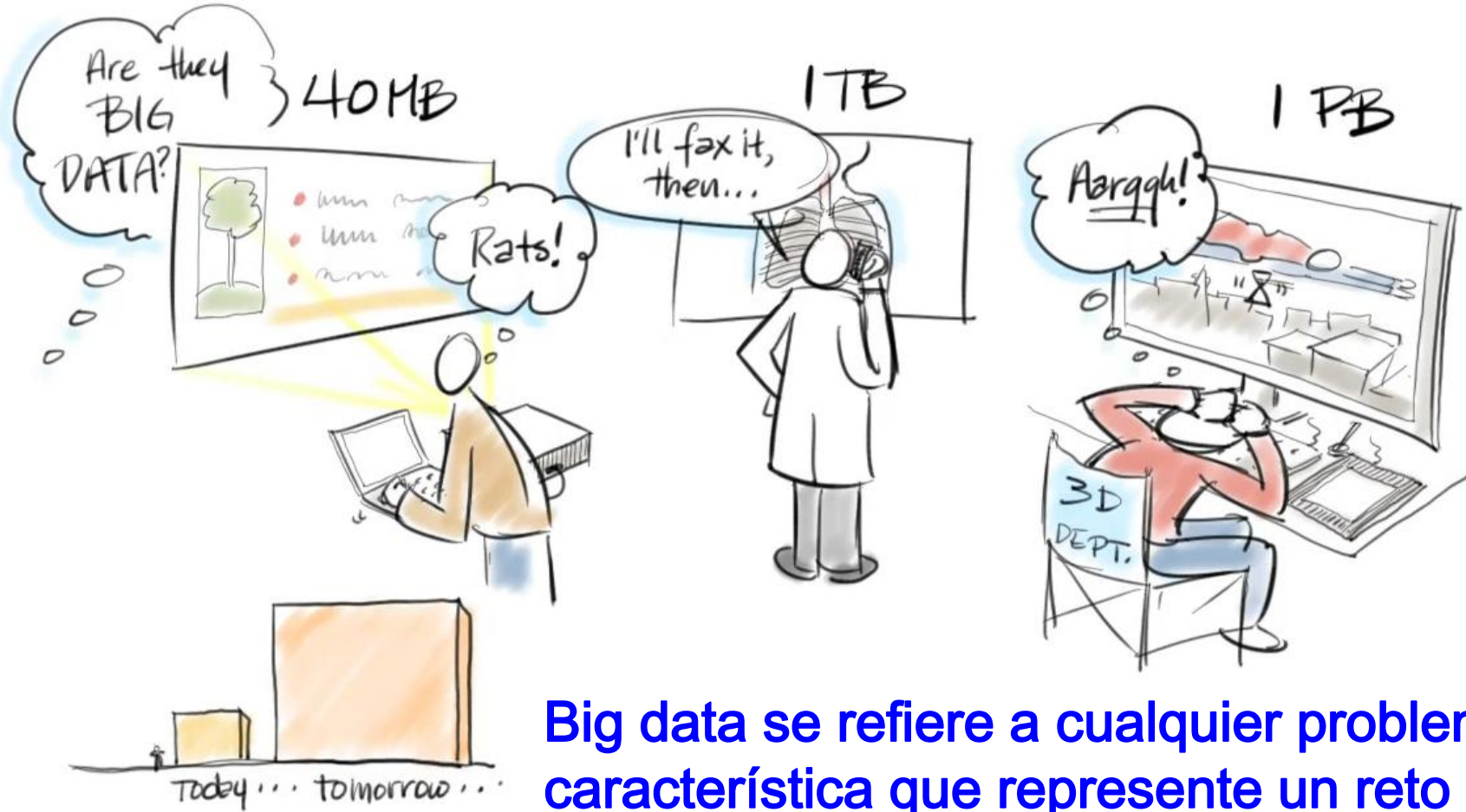
Big data es una colección de datos grande, complejos, **muy difícil de procesar a través de herramientas de gestión y procesamiento de datos tradicionales**



“**Big Data**” son datos cuyo volumen, diversidad y complejidad **requieren nueva arquitectura, técnicas, algoritmos y análisis** para gestionar y extraer valor y conocimiento oculto en ellos ...



¿Qué es Big Data?



Big data se refiere a cualquier problema o característica que represente un reto para ser procesado con aplicaciones tradicionales

¿Qué es Big Data?

¿Quién genera Big Data?



Redes sociales y multimedia
(todos generamos datos)



Instrumentos científicos
(colección de toda clase de datos)



Dispositivos móviles
(seguimiento de objetos)



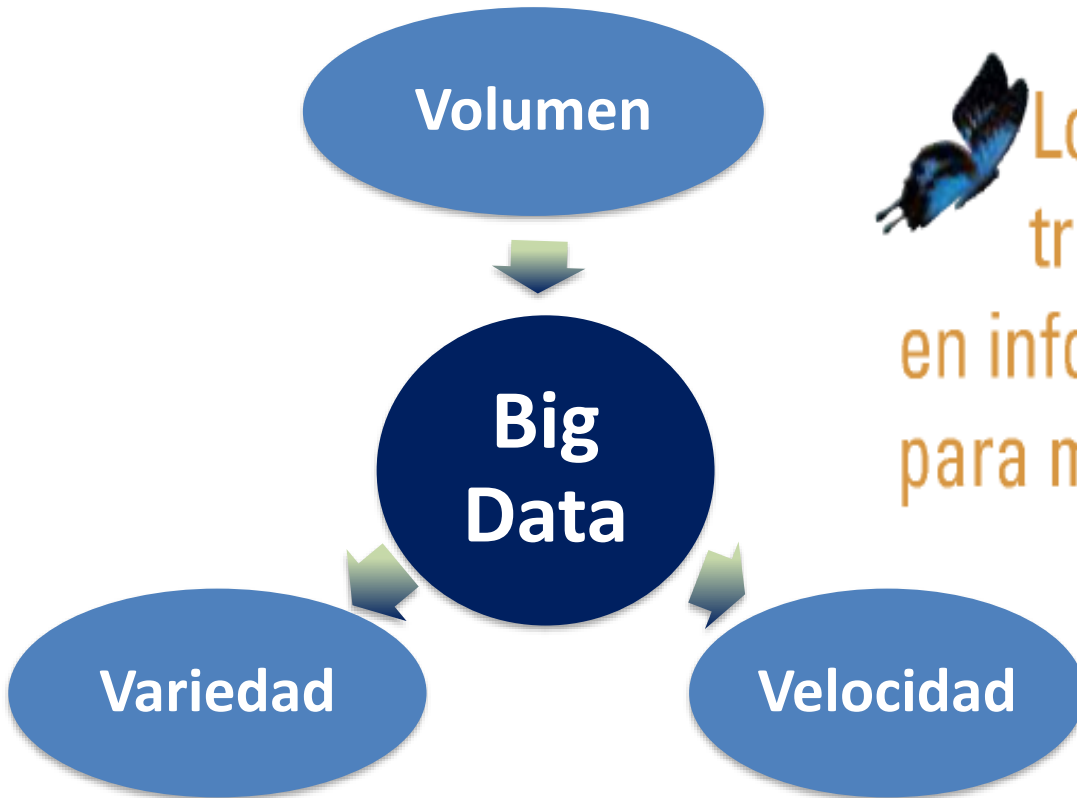
Redes de sensores
(se miden toda clase de datos)

El progreso y la innovación ya no se ven obstaculizados por la capacidad de recopilar datos, sino por la capacidad de gestionar, analizar, sintetizar, visualizar, y descubrir el conocimiento de los datos recopilados de manera oportuna y en una forma escalable

Big Data en 3 V's



FINANZAS & DESARROLLO PUBLICACIÓN TRIMESTRAL DEL FONDO MONETARIO INTERNACIONAL
Septiembre de 2016 • Volumen 53 • Número 3



Lo difícil es determinar cómo traducir esos datos brutos en información que pueda servir para mejorar el rendimiento.

Hal Varian
Economista
Jefe de Google



Doug Laney, Gartner Feb. 6, 2001

3-D Data Management: Controlling Data Volume, Velocity and Variety.

“Without Analytics, Big Data is just Noise”



FINANZAS & DESARROLLO PUBLICACIÓN TRIMESTRAL DEL FONDO MONETARIO INTERNACIONAL
Septiembre de 2016 • Volumen 53 • Número 3

Guest post by Eric Schwartzman, founder and CEO of [Comply Socially](#)



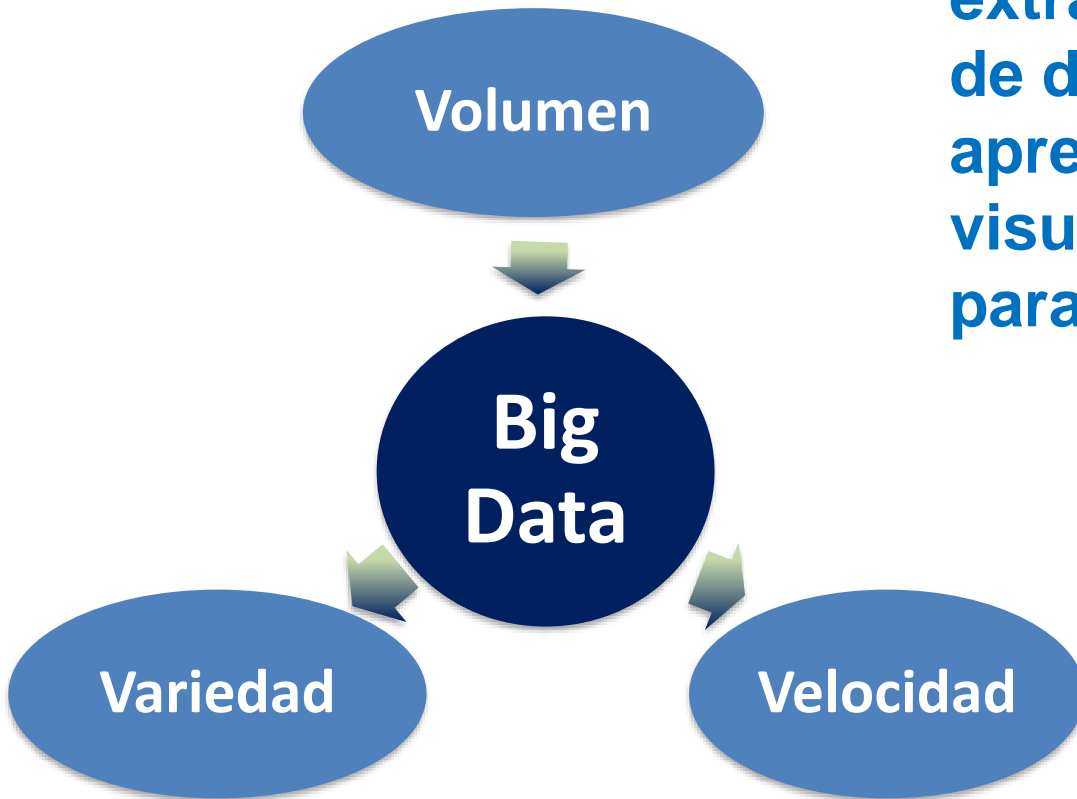
Lo difícil es determinar cómo traducir esos datos brutos en información que pueda servir para mejorar el rendimiento.



Hal Varian
Economista
Jefe de Google

<http://www.briansolis.com/2013/04/without-analytics-big-data-is-just-noise/>

Big Data en 3 V's



Ciencia de Datos (Analítica de datos): **Ámbito de ciencia y tecnología que engloba las habilidades asociadas a la extracción de conocimiento de datos (incluyendo aprendizaje automático, visualización, tecnologías para big data, ...)**



Doug Laney, Gartner Feb. 6, 2001

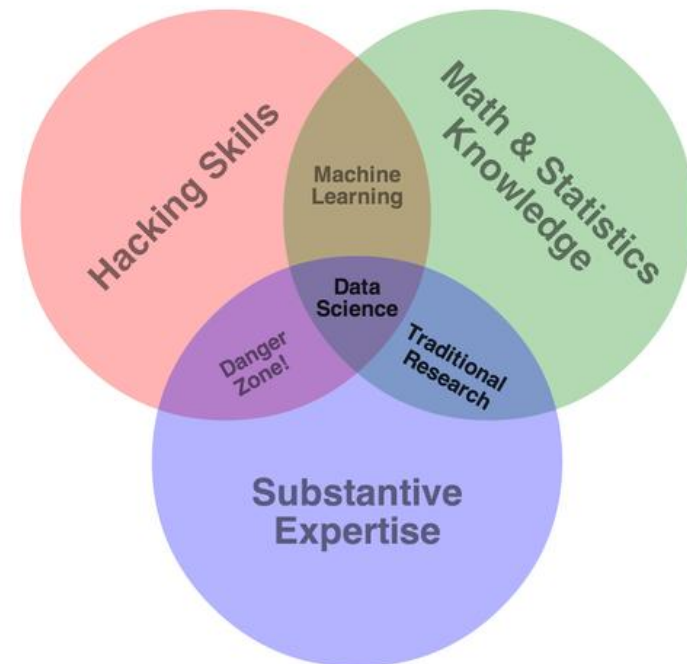
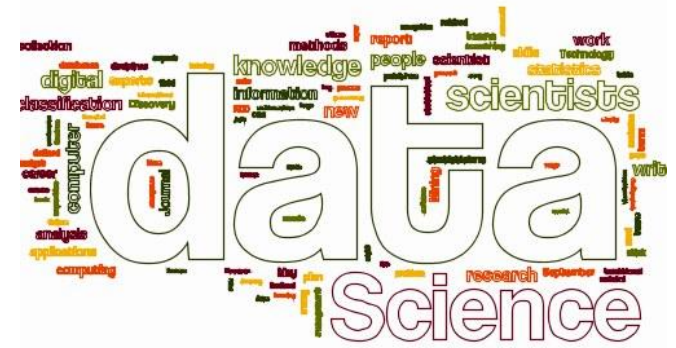
3-D Data Management: Controlling Data Volume, Velocity and Variety.



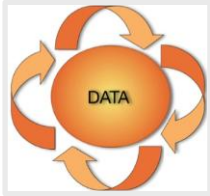
(Big) Data Science

Data Science combines the traditional scientific method with the ability to explore, learn and gain deep insight for (Big) Data

It is not just about finding patterns in data ... it is mainly about explaining those patterns



Data Science Process



Data Preprocessing

- Clean
- Sample
- Aggregate
- Imperfect data: missing, noise, ...
- Reduce dim.
- ...

> 70% time!



Data Processing

- Explore data
- Represent data
- Link data
- Learn from data
- Deliver insight
- ...



Data Analytics

- Clustering
- Classification
- Regression
- Network analysis
- Visual analytics
- Association
- ...

- ❑ **Big Data. Big Data Science**
- ❑ **¿Por qué Big Data? Google crea el Modelo de Programación MapReduce**
- ❑ Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...)
- ❑ Big Data Analytics: Librerías para Analítica de Datos en Big Data. Casos de estudio
- ❑ Algunas aplicaciones: Salud, Social Media, Identificación
- ❑ Comentarios Finales

¿Por qué Big Data?

- **Problema:** Escalabilidad de grandes cantidades de datos
- **Ejemplo:**
 - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
 - Exploración en un clúster de 1000 nodos = 33 minutos
- **Solución → Divide-Y-Vencerás**



Una sola máquina no puede gestionar grandes volúmenes de datos de manera eficiente

¿Por qué Big Data?

- **Problema:** Escalabilidad de grandes cantidades de datos
- **Ejemplo:**
 - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
 - Exploración en un clúster de 1000 nodos = 33 minutos
- **Solución → Divide-Y-Vencerás**

¿Cómo podemos procesar

1000 TB or 10000 TB?



¿Por qué Big Data?

- Escalabilidad de grandes cantidades de datos
 - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
 - Exploración en un clúster de 1000 nodos = 33 minutos

Solución → Divide-Y-Vencerás

¿Qué ocurre cuando el tamaño de los datos aumenta y los requerimientos de tiempo se mantiene?

Hace unos años: Había que aumentar los recursos de hardware (número de nodos). Esto tiene limitaciones de espacio, costes, ...

Google 2004: Paradigma **MapReduce**

MapReduce



- Escalabilidad de grandes cantidades de datos
 - Exploración 100 TB en 1 nodo @ 50 MB/sec = 23 días
 - Exploración en un clúster de 1000 nodos = 33 minutos

Solución → Divide-Y-Vencerás

MapReduce

- Modelo de programación de datos paralela
- Concepto simple, elegante, extensible para múltiples aplicaciones
- **Creado por Google (2004)**
 - Procesa 20 PB de datos por día (2004)
- **Popularizado por el proyecto de código abierto Hadoop**
 - Usado por [Yahoo!](#), [Facebook](#), [Amazon](#), ...

MapReduce



MapReduce es la aproximación más popular para Big Data

Fragmentación de datos con
Procesamiento Paralelo
+ Fusión de Modelos

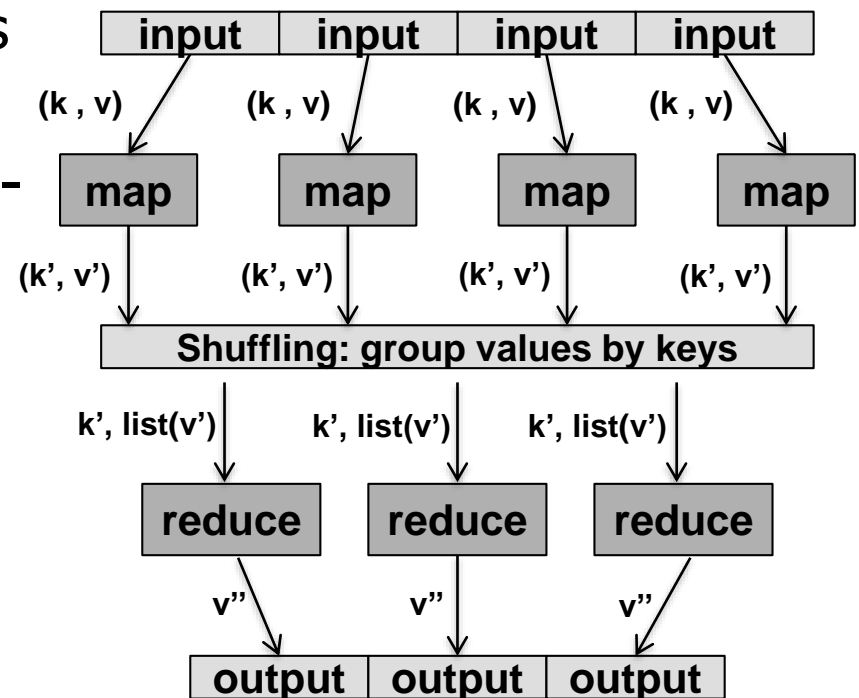


VS



MapReduce

- MapReduce es el entorno más popular para Big Data
- Basado en la estructura Valor-llave.
- Dos operaciones:
 1. **Función Map : Procesa bloques de información**
 2. **Función Reduce function: Fusiona los resultados previous de acuerdo a su llave.**
- + Una etapa intermedia de agrupamiento por llave (**Shuffling**)



$\text{map } (k, v) \rightarrow \text{list } (k', v')$
 $\text{reduce } (k', \text{list}(v')) \rightarrow v''$

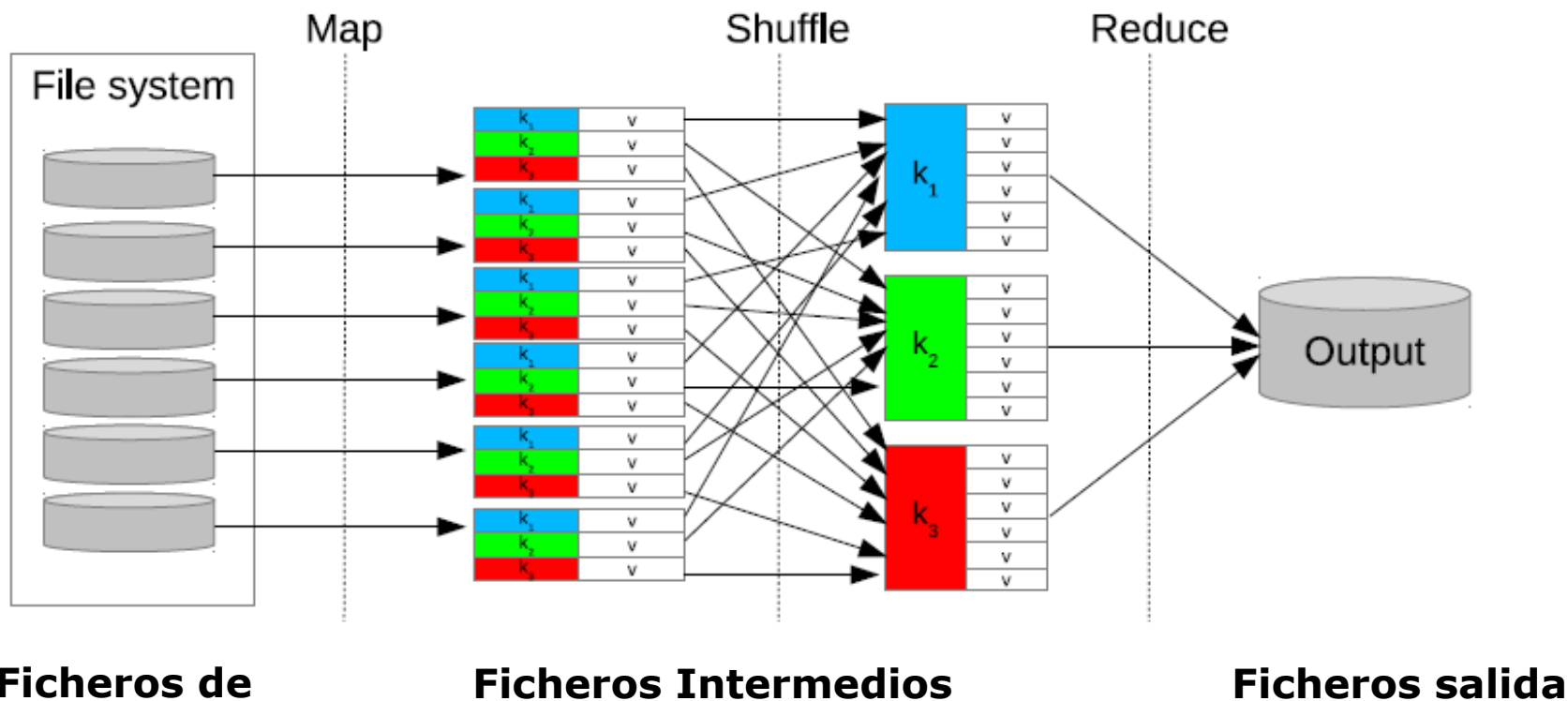
Características

- **Paralelización automática:**
 - Dependiendo del tamaño de ENTRADA DE DATOS → se crean múltiples tareas MAP
 - Dependiendo del número de intermedio <clave, valor> particiones → se pueden crear varias tareas REDUCE
- **Escalabilidad:**
 - Funciona sobre cualquier cluster de nodos/procesadores
 - Puede trabajar desde 2 a 10,000 máquinas
- **Transparencia programación**
 - Manejo de los fallos de la máquina
 - Gestión de comunicación entre máquina

MapReduce

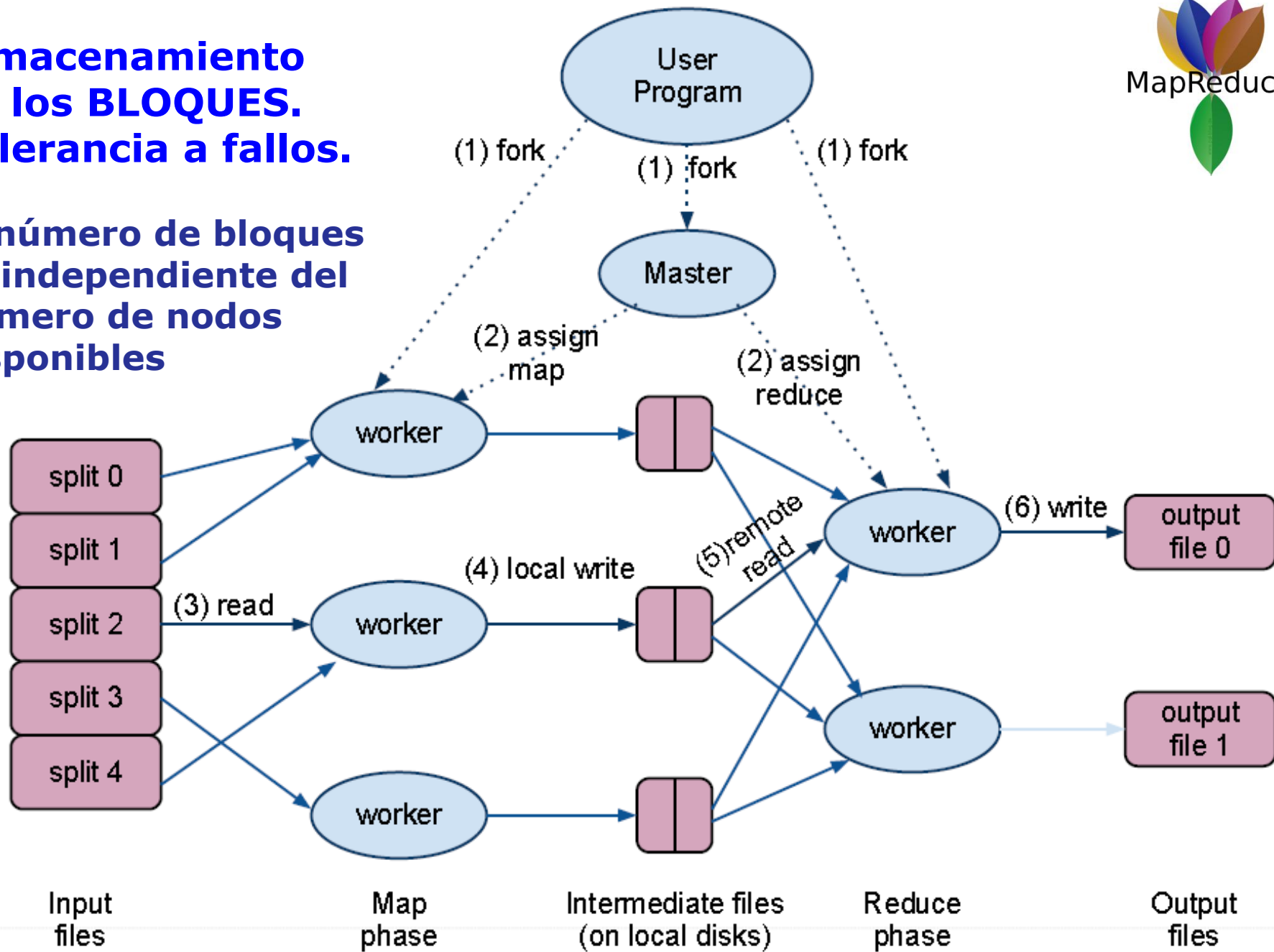


Flujo de datos en MapReduce, transparente para el programador



Almacenamiento de los BLOQUES. Tolerancia a fallos.

El número de bloques es independiente del número de nodos disponibles



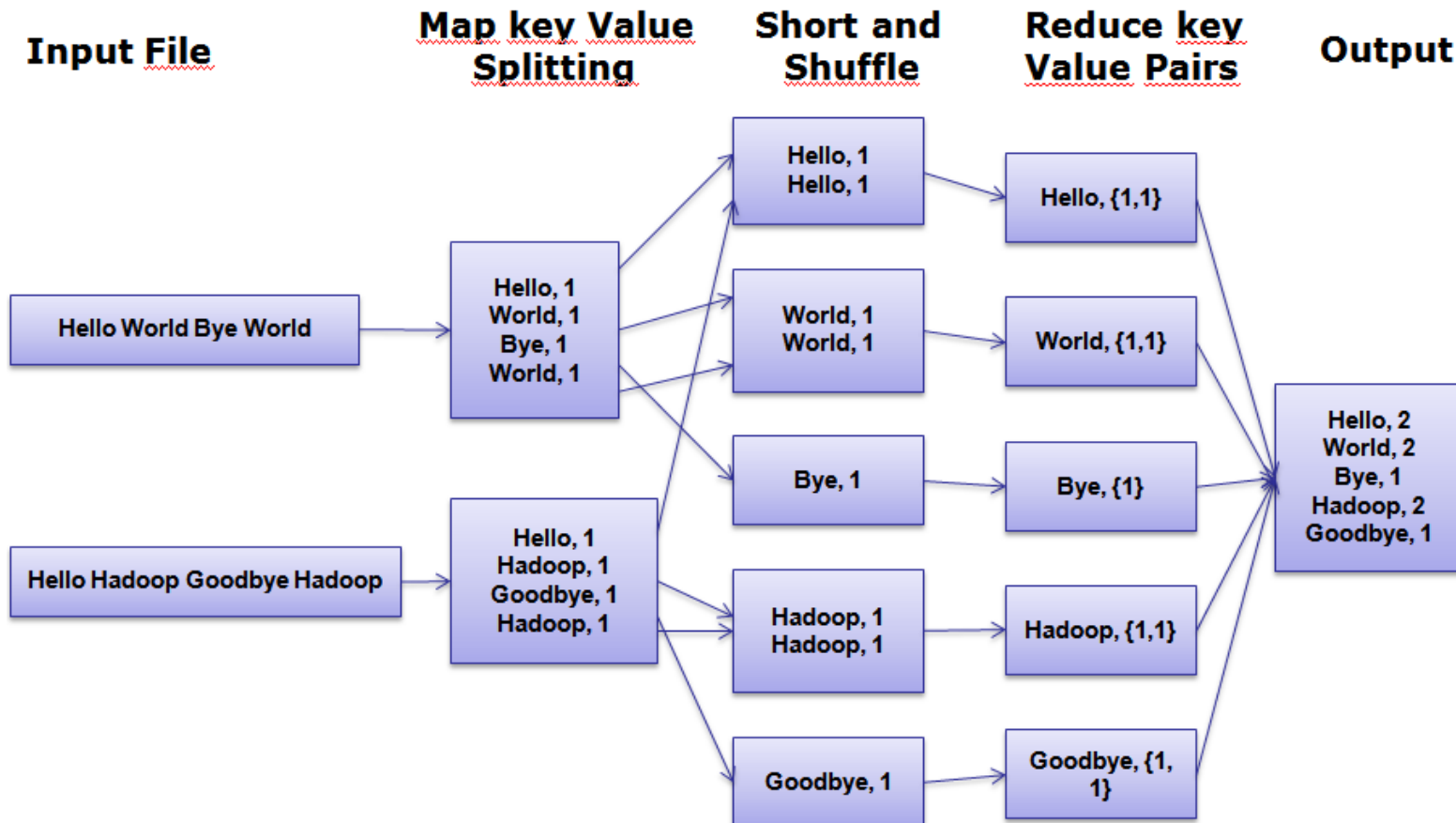
Almacenamiento con copias, normalmente $r=3$

MapReduce



Una imagen completa del proceso MapReduce

WordCount utilizando MapReduce



MapReduce



MapReduce: WordCount pseudo-code!

Pseudo-code:

map(key, value):

```
// key: document ID; value: text of document
  FOR (each word w in value)
    emit(w, 1);
```

reduce(key, value-list):

```
// key: a word; value-list: a list of integers
  result = 0;
  FOR (each count v on value-list)
    result += v;
  emit(key, result);
```

MapReduce



Resumiendo:

- **Ventaja frente a los modelos distribuidos clásicos:** El modelo de programación paralela de datos de MapReduce oculta la complejidad de la distribución y tolerancia a fallos.
- **Claves de su filosofía: Es**
 - **escalable:** *se olvidan los problemas de hardware*
 - **más barato:** *se ahorran costes en hardware, programación y administración (Commodity computing).*
- **MapReduce no es adecuado para todos los problemas, pero cuando funciona, puede ahorrar mucho tiempo**

Bibliografía: A. Fernandez, S. Río, V. López, A. Bawakid, M.J. del Jesus, J.M. Benítez, F. Herrera, **Big Data with Cloud Computing: An Insight on the Computing Environment, MapReduce and Programming Frameworks.** *WIREs Data Mining and Knowledge Discovery* 4:5 (2014) 380-409

Limitaciones

“If all you have is a hammer, then everything looks like a nail.”

MAPREDUCE
IS GOOD
ENOUGH?

ORIGINAL ARTICLE

If All You Have is a Hammer, Throw Away Everything That's Not a Nail!

Jimmy Lin

*The iSchool, University of Maryland
College Park, Maryland*



Los siguientes tipos de algoritmos son ejemplos en los que MapReduce no funciona bien:

Iterative Graph Algorithms
Gradient Descent
Expectation Maximization



Limitaciones de MapReduce

Algoritmos de grafos iterativos. Existen muchas limitaciones para estos algoritmos.

Ejemplo: Cada iteración de PageRank se corresponde a un trabajo de MapReduce.

Se han propuesto una serie de extensiones de MapReduce o modelos de programación alternativa para acelerar el cálculo iterativo:

Pregel (Google)

Pregel: A System for Large-Scale Graph Processing

Implementación: <http://www.michaelnielsen.org/ddi/pregel/>

Malewicz, G., Austern, M., Bik, A., Dehnert, J., Horn, I., Leiser, N., and Czajkowski, G. Pregel: A system for large escale graph processing. ACM SIGMOD 2010.

Limitaciones de MapReduce

MapReduce inside Google



Googlers' hammer for 80% of our data crunching

- [Large-scale web search indexing](#)
- Clustering problems for [Google News](#)
- Produce reports for popular queries, e.g. [Google Trend](#)
- Processing of [satellite imagery data](#)
- Language model processing for [statistical machine translation](#)
- Large-scale [machine learning problems](#)
- Just a plain tool to reliably spawn large number of tasks
 - e.g. parallel data backup and restore

The other 20%? e.g. [Pregel](#)



BIG DATA

Índice

- ❑ **Big Data. Big Data Science**
- ❑ **¿Por qué Big Data? Google crea el Modelo de Programación MapReduce**
- ❑ **Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...)**
- ❑ Big Data Analytics: Librerías para Analítica de Datos en Big Data. Casos de estudio
- ❑ Algunas aplicaciones: Salud, Social Media, Identificación
- ❑ Comentarios Finales

Hadoop



**Hadoop es una
implementación de
código abierto del
paradigma
computacional
MapReduce**

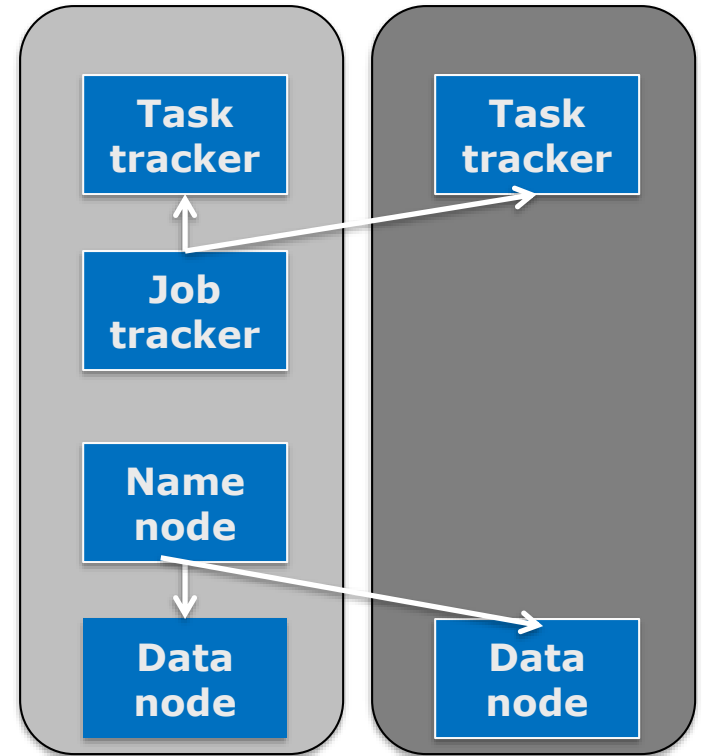
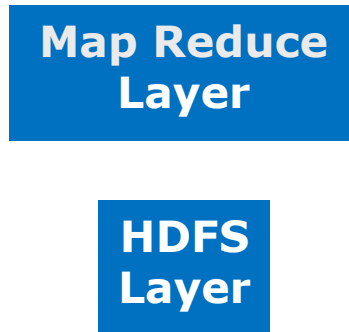
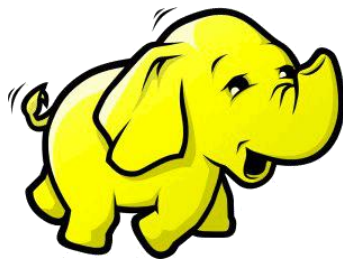


<http://hadoop.apache.org/>

Hadoop



Hadoop Distributed File System (HDFS) es un sistema de archivos distribuido, escalable y portátil escrito en **Java** para el framework Hadoop



Creado por **Doug Cutting** (chairman of board of directors of the Apache Software Foundation, 2010)

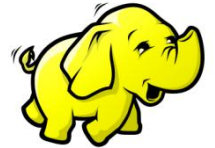


<http://hadoop.apache.org/>

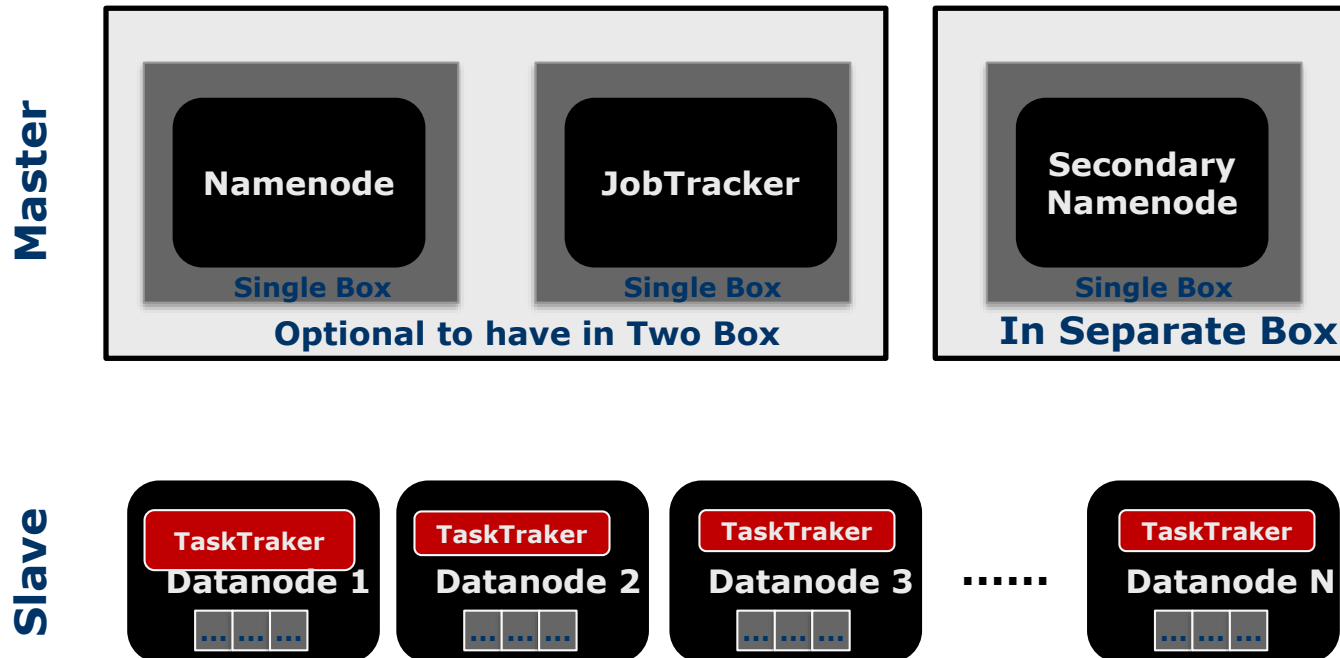
Hadoop



Hadoop: A master/slave architecture



- **Master:** NameNode, JobTracker
- **Slave:** {DataNode, TaskTraker}, ..., {DataNode, TaskTraker}



Hadoop



<http://sortbenchmark.org/>

Primer hito de Hadoop: July 2008 - Hadoop Wins Terabyte Sort Benchmark

Uno de los grupos de Yahoo Hadoop ordenó 1 terabyte de datos en 209 segundos, superando el récord anterior de 297 segundos en la competición anual de ordenación de un terabyte (Daytona).

Esta es la primera vez que un programa en Java de código abierto ganó la competición.

2008, 3.48 minutes

Hadoop

910 nodes x (4 dual-core processors, 4 disks, 8 GB memory)
Owen OMalley, Yahoo

2007, 4.95 min

TokuSampleSort

tx2500 disk cluster
400 nodes x (2 processors, 6-disk RAID, 8 GB memory)
Bradley C. Kuzmaul , MIT

Daytona

2013, 1.42 TB/min

Hadoop

102.5 TB in 4,328 seconds
2100 nodes x
(2 2.3Ghz hexcore Xeon E5-2630, 64 GB memory, 12x3TB disks)
Thomas Graves
Yahoo! Inc.

Gray

<http://developer.yahoo.com/blogs/hadoop/hadoop-sorts-petabyte-16-25-hours-terabyte-62-422.html>

Ecosistema Hadoop



El proyecto **Apache Hadoop** incluye los módulos:

Hadoop Common: Las utilidades comunes que apoyan los otros módulos de Hadoop.

Hadoop Distributed File System (HDFS): El sistema de ficheros que proporciona el acceso

Hadoop YARN: Marco para el manejo de recursos de programación y grupo de trabajo.

Hadoop MapReduce: Un sistema de basado en YARN o para el procesamiento en paralelo de grandes conjuntos de datos.

<http://hadoop.apache.org/>

Ecosistema Apache Hadoop incluye más de 150 proyectos:

Avro: Un sistema de serialización de datos.

Cassandra: Una base de datos escalable multi-master sin puntos individuales y fallo

Chukwa: Un sistema de recogida de datos para la gestión de grandes sistemas distribuidos.

Hbase: Una base de datos distribuida, escalable que soporta estructurado de almacenamiento de datos para tablas de gran tamaño.

Hive: Un almacén de datos que proporciona el Resumen de datos para tablas de gran tamaño.

Pig: Lenguaje para la ejecución de alto nivel de flujo de datos para computación paralela.

Tez: Sustituye al modelo "MapShuffleReduce" por un flujo de ejecución con grafos acíclico dirigido (DAG)

Giraph: Procesamiento iterativo de grafos

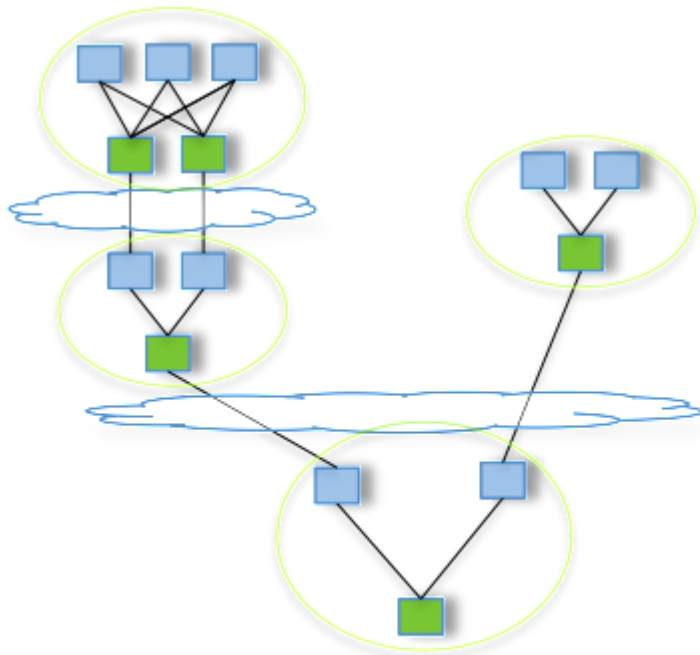
Mahout: Aprendizaje automático escalable (biblioteca de minería de datos)

Recientemente: **Apache Spark**

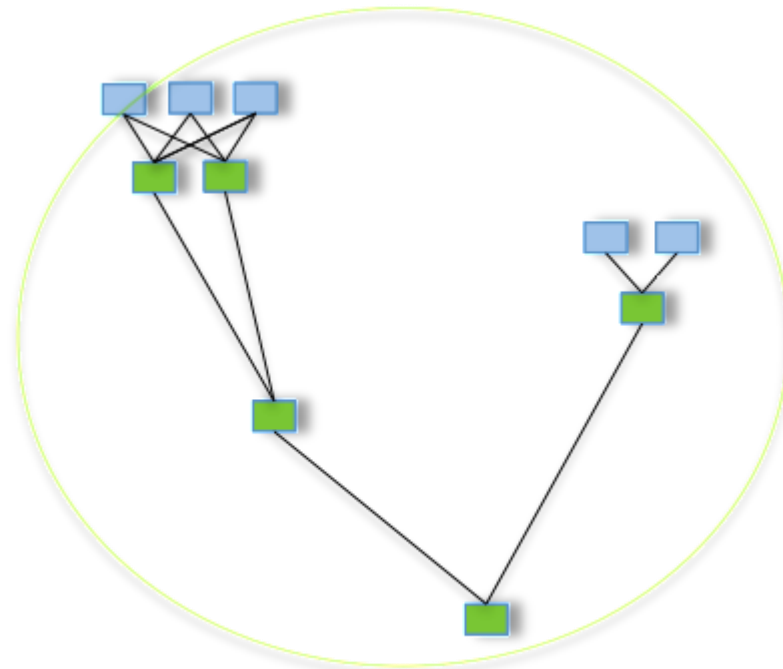


Limitaciones de MapReduce

Procesos con flujos acíclicos de procesamiento de datos



Pig/Hive - MR



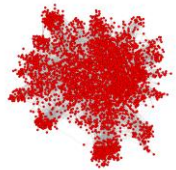
Pig/Hive - Tez

<http://tez.apache.org/>

Limitaciones de MapReduce: Nuevas herramientas



GIRAPH (APACHE Project)
(<http://giraph.apache.org/>)
Procesamiento iterativo de grafos



GPS - A Graph Processing System, (Stanford)
<http://infolab.stanford.edu/gps/>
para Amazon's EC2



Distributed GraphLab (Carnegie Mellon Univ.)
<https://github.com/graphlab-code/graphlab>
Amazon's EC2



Spark (UC Berkeley) (Apache Foundation)
<http://spark.incubator.apache.org/research.html>



Twister (Indiana University)
<http://www.iterativemapreduce.org/>
Clusters propios



PrIter (University of Massachusetts Amherst, Northeastern University-China)
<http://code.google.com/p/priter/>
Cluster propios y Amazon EC2 cloud



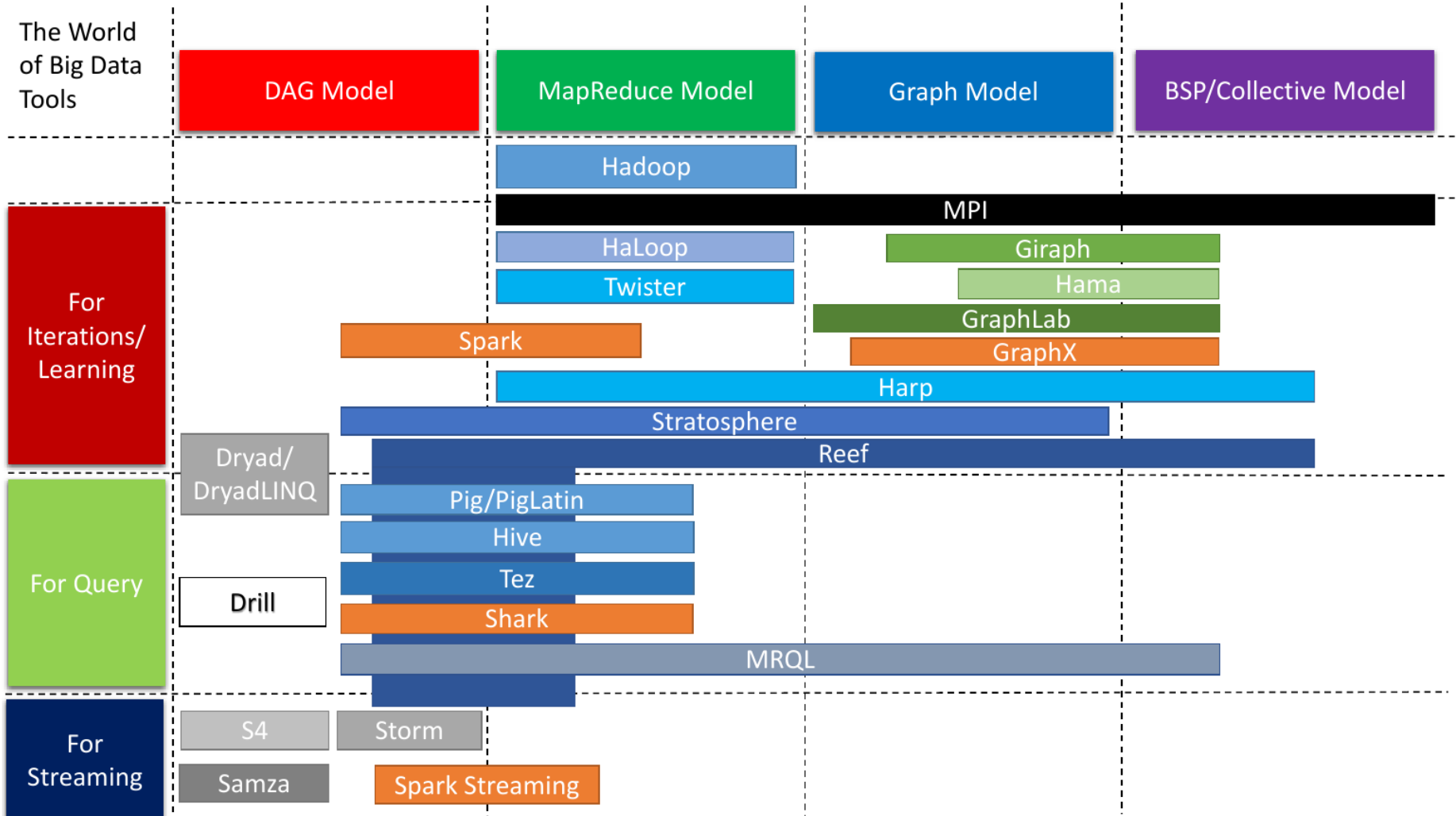
HaLoop (University of Washington)
<http://clue.cs.washington.edu/node/14>
<http://code.google.com/p/haloop/>
Amazon's EC2

GPU based platforms

**Mars
GreX
GPMR**



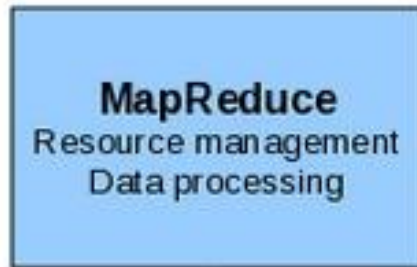
Limitaciones de MapReduce: Nuevas herramientas



Evolución de Hadoop

Evolución de Hadoop

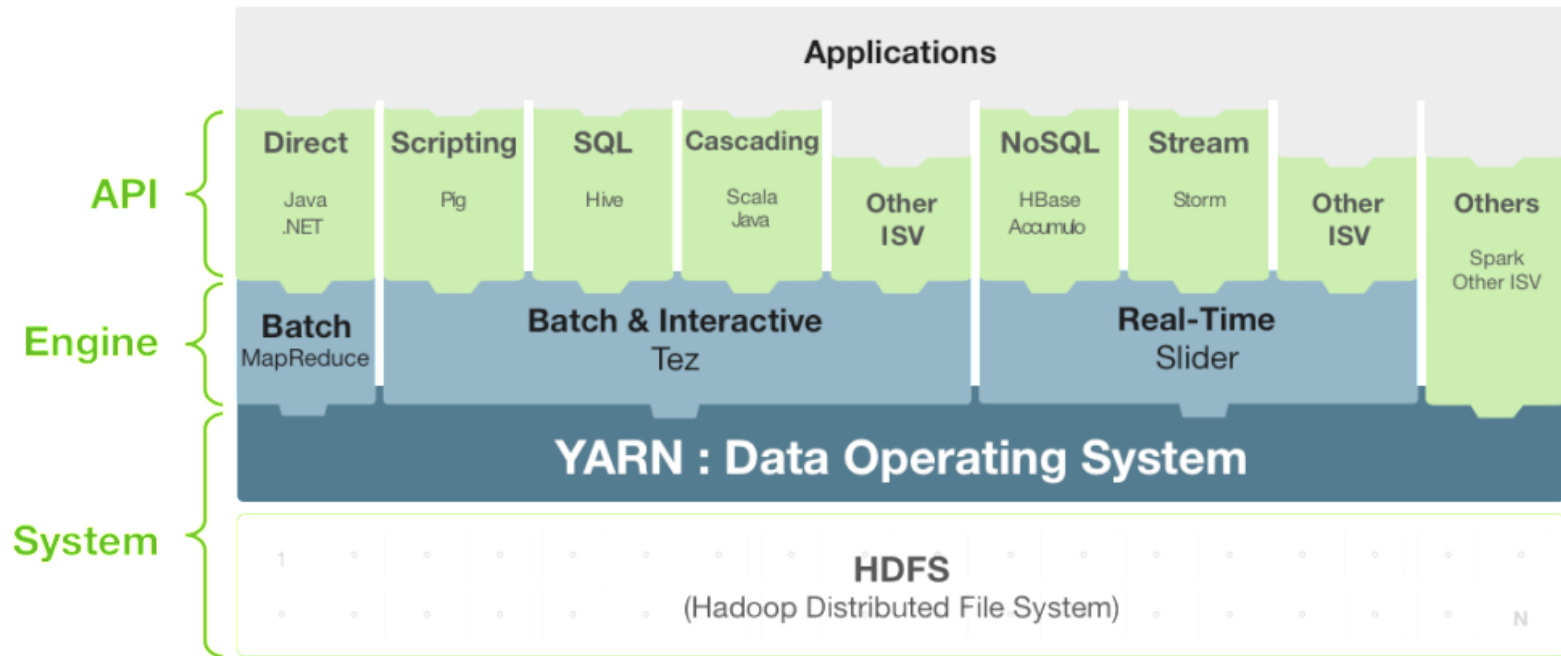
Hadoop V1



Hadoop V2

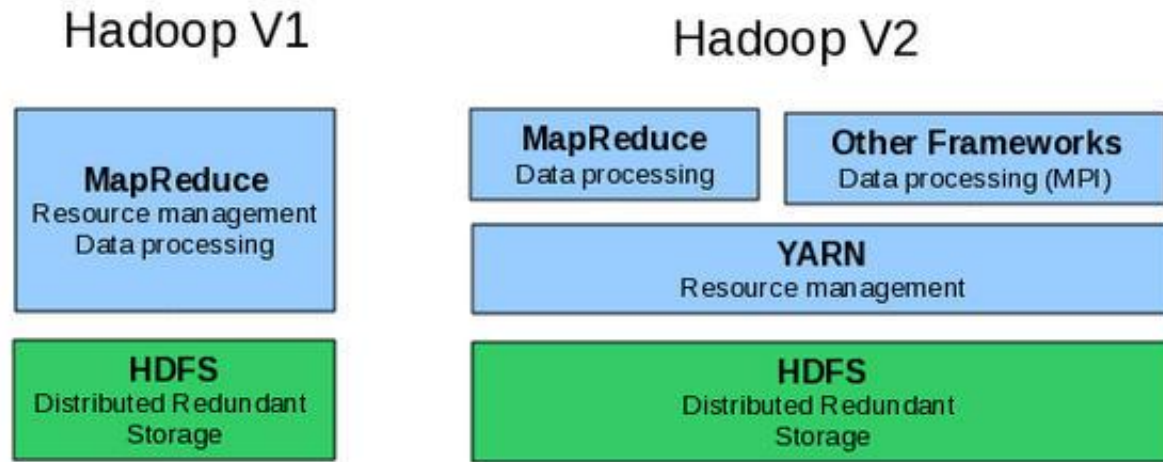


Apache Hadoop YARN



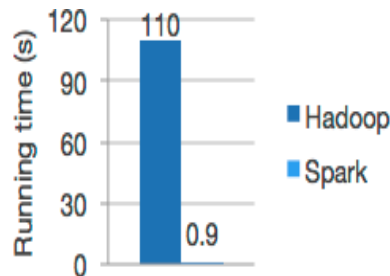
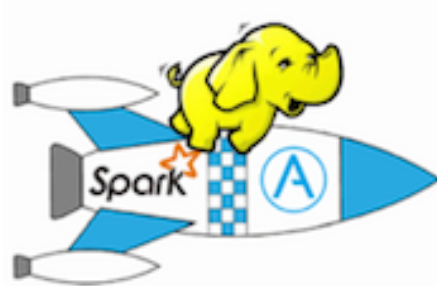
Apache Hadoop YARN es el sistema operativo de datos de Hadoop 2, responsable de la gestión del acceso a los recursos críticos de Hadoop. YARN permite al usuario interactuar con todos los datos de múltiples maneras al mismo tiempo, haciendo de Hadoop una verdadera plataforma de datos multi-uso y lo que le permite tomar su lugar en una arquitectura de datos moderna.

Apache Spark



<https://spark.apache.org/>

**Enfoque InMemory
HDFS Hadoop + SPARK**



**Fast and expressive
cluster computing
system compatible
with Apache Hadoop**

Apache Spark (Birth 2009-2010)



Fast and Expressive Cluster Computing Engine Compatible with Apache Hadoop

Up to **10X** faster on disk,
100X in memory

Efficient

- General execution graphs
- In-memory storage

2-5X less code

Usable

- Rich APIs in Java, Scala, Python
- Interactive shell

Apache Spark

Spark Programming Model

KEY Concept: RDD (Resilient Distributed Datasets)

Write programs in terms of operations on distributed data sets

- Collection of objects spread across a cluster, stored in RAM or on Disk
- Built through parallel transformations on distributed datasets
- An RDD is a fault-tolerant collection of elements that can be operated on in parallel.
- There are two ways to create RDDs:

Parallelizing an existing collection in your driver program

Referencing a dataset in an external storage system, such as a shared filesystem, HDFS, Hbase.

- *Can be cached for future reuse*
- Built through parallel transformations on distributed datasets
- RDD operations: transformations and actions

Transformations (e.g. map, filter, groupBy)...

(Lazy operations to build RDSs from other RDSs)

Actions (eg. Count, collect, save ...)

(Return a result or write it a storage)

Apache Spark

Spark Operations

| | | |
|---|---|---|
| Transformations (define a new RDD) | map filter sample groupByKey reduceByKey sortByKey | flatMap union join cogroup cross mapValues |
| Actions (return a result to driver program) | | collect reduce count save lookupKey |

Zaharia-2012- Zaharia M, Chowdhury M, Das T, Dave A, Ma J, McCauley M, Franklin MJ, Shenker S, Stoica I.
Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing.

In: *9th USENIX Conference on Networked Systems Design and Implementation, San Jose, CA, 2012, 1–14.*

Apache Spark

- RDDs allow us to express different programming models:
 - MapReduce.
 - Iterative MapReduce. It can implement HaLoop or Twister systems.
 - Stream processing.
 - Iterative graph applications (Google's Pregel).
 - SQL.
 - ...
- It provides us more flexibility to design scalable ML tools.

Zaharia-2012- Zaharia M, Chowdhury M, Das T, Dave A, Ma J, McCauley M, Franklin MJ, Shenker S, Stoica I.
Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing.

In: *9th USENIX Conference on Networked Systems Design and Implementation, San Jose, CA, 2012*, 1–14.

Apache Spark

DataFrames

A DataFrame is a distributed collection of data organized into named columns. It is conceptually equivalent to a table in a relational database or a data frame in R/Python, but with richer optimizations under the hood. DataFrames can be constructed from a wide array of [sources](#) such as: structured data files, tables in Hive, external databases, or existing RDDs.

The DataFrame API is available in [Scala](#), [Java](#), [Python](#), and [R](#).

Datasets

A Dataset is a new experimental interface added in Spark 1.6 that tries to provide the benefits of RDDs (strong typing, ability to use powerful lambda functions) with the benefits of Spark SQL's optimized execution engine. A Dataset can be [constructed](#) from JVM objects and then manipulated using functional transformations (map, flatMap, filter, etc.).

The unified Dataset API can be used both in [Scala](#) and [Java](#). Python does not yet have support for the Dataset API, but due to its dynamic nature many of the benefits are already available (i.e. you can access the field of a row by name naturally `row.columnName`). Full python support will be added in a future release.

Zaharia-2012- Zaharia M, Chowdhury M, Das T, Dave A, Ma J, McCauley M, Franklin MJ, Shenker S, Stoica I.
Resilient distributed datasets: a fault-tolerant abstraction for in-memory cluster computing.

In: *9th USENIX Conference on Networked Systems Design and Implementation, San Jose, CA, 2012, 1–14.*

Apache Spark

| | Hadoop World Record | Spark 100 TB * |
|---------------------------------|--------------------------|-------------------|
| Data Size | 102.5 TB | 100 TB |
| Elapsed Time | 72 mins | 23 mins |
| # Nodes | 2100 | 206 |
| # Cores | 50400 | 6592 |
| # Reducers | 10,000 | 29,000 |
| Rate | 1.42 TB/min | 4.27 TB/min |
| Rate/node | 0.67 GB/min | 20.7 GB/min |
| Sort Benchmark Daytona Rules | Yes | Yes |
| Environment | dedicated data center | EC2 (i2.8xlarge) |

October 10, 2014

Using Spark on 206 EC2 nodes, we completed the benchmark in 23 minutes. This means that Spark sorted the same data 3X faster using 10X fewer machines. All the sorting took place on disk (HDFS), without using Spark's in-memory cache.

Apache Spark

November 5, 2014

Daytona

2013, 1.42 TB/min

Hadoop

102.5 TB in 4,328 seconds
2100 nodes x
(2 2.3Ghz hexcore Xeon E5-2630, 64 GB memory, 12x3TB disks)

Thomas Graves
Yahoo! Inc.

Gray

Daytona

2-way tie:
2014, 4.35 TB/min

TritonSort

100 TB in 1,378 seconds
186 Amazon EC2 i2.8xlarge nodes x
(32 vCores - 2.50Ghz Intel Xeon E5-2670 v2, 244GB memory,
8x800 GB SSD)

Michael Conley, Amin Vahdat,
George Porter
University of California, San Diego

2014, 4.27 TB/min

Apache Spark

100 TB in 1,406 seconds
207 Amazon EC2 i2.8xlarge nodes x
(32 vCores - 2.5Ghz Intel Xeon E5-2670 v2, 244GB memory,
8x800 GB SSD)

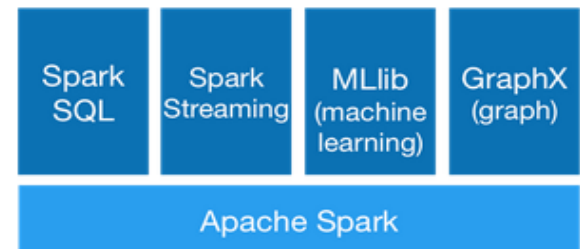
Reynold Xin, Parviz Deyhim, Xiangrui Meng,
Ali Ghodsi, Matei Zaharia
Databricks

Gray

Apache Spark



Ecosistema Apache Spark

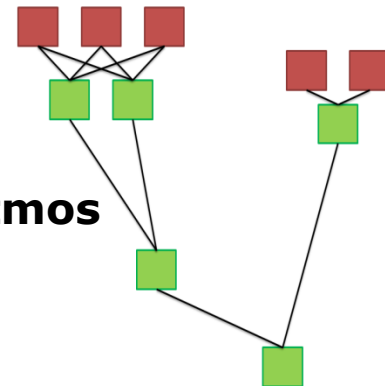



- **Big Data "in-memory"**. Spark permite realizar trabajos paralelizados totalmente en memoria, lo cual reduce mucho los tiempos de procesamiento. Sobre todo si se trata de unos procesos iterativos. En el caso de que algunos datos no quepan en la memoria, Spark seguirá trabajando y usará el disco duro para volcar aquellos datos que no se necesitan en este momento (Hadoop **"commodity hardware"**).

KEY Concept: RDD (Resilient Distributed Datasets)

Write programs in terms of operations on distributed data sets

- **Esquema de computación más flexible que MapReduce.** Permite la flujos acíclicos de procesamiento de datos, algoritmos iterativos
- **Spark ofrece una API para Java, Python y Scala**



-  Spark / Wiki Homepage
Powered By Spark

Creado por Andy Konwinski, modificado por última vez por Reynold Xin el sep 08, 2015

**Databricks, Groupon, eBay inc.,
Amazon, Hitachi, Nokia, Yahoo!, ...**

<https://cwiki.apache.org/confluence/display/SPARK/Powered+By+Spark>

Hadoop



¿Cómo accedo a una plataforma Hadoop?

Plataformas Cloud con instalación de Hadoop

Amazon Elastic Compute Cloud (Amazon EC2)
<http://aws.amazon.com/es/ec2/>



Windows Azure™
Windows Azure

<http://www.windowsazure.com/>



Instalación en un cluster Ejemplo ATLAS, infraestructura del grupo SCI²S

Cluster ATLAS: 4 super servers from Super Micro Computer Inc. (4 nodes per server)
The features of each node are:

- ❑ Microprocessors: 2 x Intel Xeon E5-2620 (6 cores/12 threads, 2 GHz, 15 MB Cache)
- ❑ RAM 64 GB DDR3 ECC 1600MHz, Registered
- ❑ 1 HDD SATA 1TB, 3Gb/s; (system)
- ❑ 1 HDD SATA 2TB, 3Gb/s; (distributed file system)



cloudera

Distribución que ofrece Cloudera para Hadoop

<http://www.cloudera.com/content/cloudera/en/why-cloudera/hadoop-and-big-data.html>



Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...) (Una instantánea)

Big Data is high-volume, high-velocity and high-variety information assets that demand cost-effective, innovative forms of information processing for enhanced insight and decision making.

There are two main issues related to Big Data:

1. **Database/storage frameworks: to write, read, and manage data.**
2. **Computational models: to process and analyze data.**

Recently, there are the following Big Data frameworks:

1. **Storage frameworks: Google File System (GFS), Hadoop Distributed File Systems (HDFS).**
2. **Computational models: MapReduce (Apache Hadoop), Resilient Distributed Datasets (RDD by Apache Spark).**



Big Data: Tecnología y Cronología

2001-2010
2010-2015

(Una instantánea)

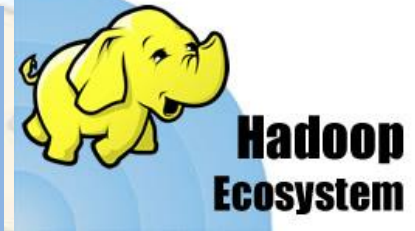


BIG DATA

2001
3V's Gartner
Doug Laney



2004
MapReduce
Google
Jeffrey Dean



2009-2013 Flink
TU Berlin
Flink Apache (Dec. 2014)
Volker Markl



Big Data



2008
Hadoop
Yahoo!
Doug Cutting

2010-2017:
Big Data Analytics:
Mahout, MLLib,
...
Hadoop Ecosystem
Aplicaciones
Nuevas Tecnologías



2010 Spark
U Berkeley
Apache Spark Feb. 2014
Matei Zaharia



BIG DATA

Índice

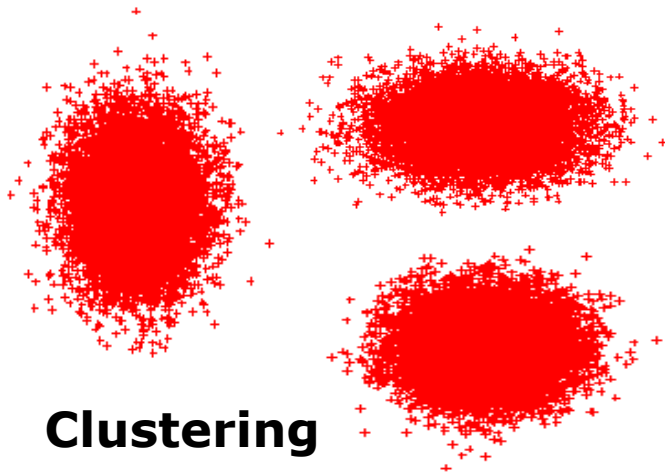
- ❑ Big Data. Big Data Science
- ❑ ¿Por qué Big Data? Google crea el Modelo de Programación MapReduce
- ❑ Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...)
- ❑ **Big Data Analytics: Librerías para Analítica de Datos en Big Data. Casos de estudio**
- ❑ Algunas aplicaciones: Salud, Social Media, Identificación
- ❑ Comentarios Finales

Big Data Analytics

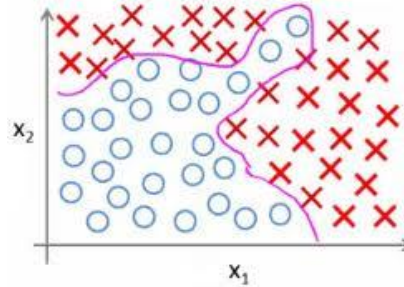
- ❑ **Big Data Analytics: Escenario**
- ❑ **Big Data Analytics: Tools**
(Mahout, MLLib, H2O, Deep Learning)
- ❑ **Caso de estudio: Random Forest**
- ❑ **Aprendizaje no supervisado: Caso de estudio K-Means**
- ❑ **Big Data Analytics: 3 Comentarios finales.**
 - ❑ Without Analytics, Big data is just noise
 - ❑ Big data preprocessing: Es necesario
 - ❑ Los expertos en Ciencia de Datos son necesarios en el uso de herramientas de Analytics y Big Data.

Big Data Analytics

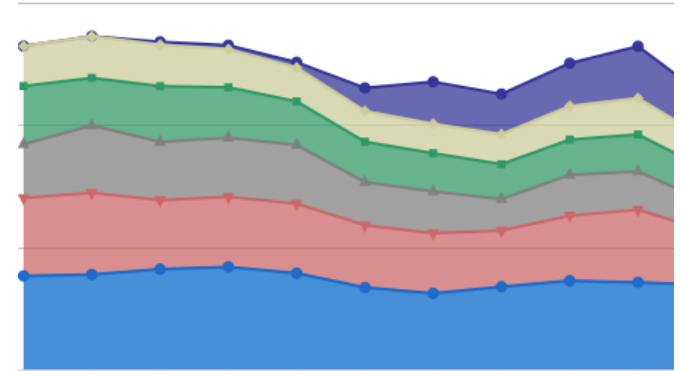
Potenciales escenarios:



Clustering

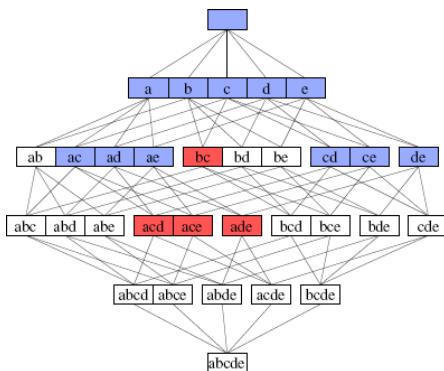


Classification



**Real Time Analytics/
Big Data Streams**

Association



**Recommendation
Systems**



**Social Media Mining
Social Big Data**

Big Data Analytics: **Tools**

| Generation | 1st Generation | 2nd Generation | 3rd Generation |
|---------------------------------|-------------------------------|--|---|
| Examples | SAS, R, Weka, SPSS, KEEL | Mahout, Pentaho, Cascading | Spark, Hadoop, GraphLab, Pregel, Giraph, ML over Storm |
| Scalability | Vertical | Horizontal (over Hadoop) | Horizontal (Beyond Hadoop) |
| Algorithms Available | Huge collection of algorithms | Small subset: sequential logistic regression, linear SVMs, Stochastic Gradient Decendent, k-means clustsering, Random forest, etc. | Much wider: CGD, ALS, collaborative filtering, kernel SVM, matrix factorization, Gibbs sampling, etc. |
| Algorithms Not Available | Practically nothing | Vast no.: Kernel SVMs, Multivariate Logistic Regression, Conjugate Gradient Decendent, ALS, etc. | Multivariate logistic regression in general form, k-means clustering, etc. – Work in progress to expand the set of available algorithms |
| Fault-Tolerance | Single point of failure | Most tools are FT, as they are built on top of Hadoop | FT: HaLoop, Spark Not FT: Pregel, GraphLab, Giraph |

Big Data Analytics: Tools

Mahout



| Classification | Single Machine | MapReduce |
|---|----------------|-----------|
| Logistic Regression - trained via SGD | x | |
| Naive Bayes / Complementary Naive Bayes | | x |
| Random Forest | | x |
| Hidden Markov Models | x | |
| Multilayer Perceptron | x | |

MLlib types, algorithms and utilities

This lists functionality included in `spark.mllib`, the main MLlib API.

- Data types
- Basic statistics
 - summary statistics
 - correlations
 - stratified sampling
 - hypothesis testing
 - random data generation
- Classification and regression
 - linear models (SVMs, logistic regression, linear regression)
 - naive Bayes
 - decision trees
 - ensembles of trees (Random Forests and Gradient-Boosted Trees)
 - isotonic regression
- Collaborative filtering
 - alternating least squares (ALS)
- Clustering
 - k-means
 - Gaussian mixture
 - power iteration clustering (PIC)
 - latent Dirichlet allocation (LDA)
 - streaming k-means
- Dimensionality reduction
 - singular value decomposition (SVD)
 - principal component analysis (PCA)
- Feature extraction and transformation
- Frequent pattern mining
 - FP-growth
- Optimization (developer)
 - stochastic gradient descent
 - limited-memory BFGS (L-BFGS)
- PMML model export

MLlib



<https://spark.apache.org/mllib/>

Mahout



Scalable machine learning
and data mining



Apache Mahout has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining



Biblioteca de código abierto

Currently Mahout supports mainly three use cases: Recommendation mining takes users' behavior and from that tries to find items users might like. Clustering takes e.g. text documents and groups them into groups of topically related documents. Classification learns from existing categorized documents what documents of a specific category look like and is able to assign unlabelled documents to the (hopefully) correct category.

<http://mahout.apache.org/>

Latest release version 0.9 has

- User and Item based recommenders
- Matrix factorization based recommenders
- K-Means, Fuzzy K-Means clustering
- Latent Dirichlet Allocation
- Singular Value Decomposition
- Logistic regression classifier
- (Complementary) Naive Bayes classifier
- Random forest classifier
- High performance java collections
- A vibrant community



Historia

25 July 2013 - Apache Mahout 0.8 released

Visit our [release notes](#) page for details.

16 June 2012 - Apache Mahout 0.7 released

Visit our [release notes](#) page for details.

6 Feb 2012 - Apache Mahout 0.6 released

Visit our [release notes](#) page for details.

9 Oct 2011 - Mahout in Action released

The book Mahout in Action is available in print. Sean Owen, Robin Anil, Ted Dunning and Ellen Friedman thank the community (especially those who were reviewers) for input during the process and hope it is enjoyable.

Find it at your favorite bookstore, or [order print and eBook copies from Manning](#) – use discount code "mahout37" for 37% off.



Historia

1 February 2014 - Apache Mahout 0.9 released

Apache Mahout has reached version 0.9. All developers are encouraged to begin using version 0.9. Highlights include:

- New and improved Mahout website based on Apache CMS - MAHOUT-1245
- Early implementation of a Multi Layer Perceptron (MLP) classifier - MAHOUT-1265
- Scala DSL Bindings for Mahout Math Linear Algebra. See this [blogpost](#) and MAHOUT-1297
- Recommenders as Search. See [<https://github.com/pferrel/solr-recommender>] and MAHOUT-1288
- Support for easy functional Matrix views and derivatives - MAHOUT-1300
- JSON output format for ClusterDumper - MAHOUT-1343
- Enabled randomised testing for all Mahout modules using Carrot RandomizedRunner - MAHOUT-1345
- Online Algorithm for computing accurate Quantiles using 1-dimensional Clustering - See this [pdf](#) and MAHOUT-1361
- Upgrade to Lucene 4.6.1 - MAHOUT-1364

Changes in 0.9 are detailed in the [release notes](#).

The following algorithms that were marked deprecated in 0.8 have been removed in 0.9:

- Switched LDA implementation from Gibbs Sampling to Collapsed Variational Bayes
- Meanshift - removed due to lack of actual usage and support
- MinHash - removed due to lack of actual usage and support
- Winnow - removed due to lack of actual usage and support
- Perceptron - removed due to lack of actual usage and support
- Slope One - removed due to lack of actual usage
- Distributed Pseudo recommender - removed due to lack of actual usage
- TreeClusteringRecommender - removed due to lack of actual usage

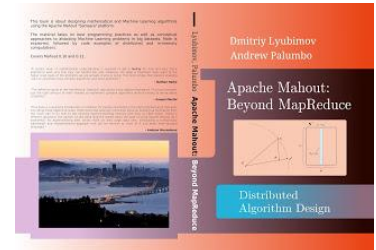
Mahout



Historia

12 March 2016 - Apache Mahout 0.11.2 released

23 February 2016 - New Apache Mahout Book - "Apache Mahout: Beyond MapReduce" by D.Lyubimov and A.Palumbo released. See [Mahout "Samsara" Book Is Out](#)



Mahout News

25 April 2014 - Goodbye MapReduce

The Mahout community decided to move its codebase onto modern data processing systems that offer a richer programming model and more efficient execution than Hadoop MapReduce. **Mahout will therefore reject new MapReduce algorithm implementations from now on.** We will however keep our widely used MapReduce algorithms in the codebase and maintain them.

We are building our future implementations on top of a DSL for linear algebraic operations which has been developed over the last months. Programs written in this DSL are automatically optimized and executed in parallel on Apache Spark.

Furthermore, there is an experimental contribution undergoing which aims to integrate the h2o platform into Mahout.

Mahout

Algoritmos

Es una buena librería para introducirse en el uso de MapReduce sobre Hadoop



| | Single Machine | MapReduce | Spark | H2O | Flink |
|--|----------------|-----------|-------|-----|-----------------------|
| Mahout Math-Scala Core Library and Scala DSL | | | | | |
| Mahout Distributed BLAS. Distributed Row Matrix API with R and Matlab like operators. Distributed ALS, SPCA, SSVD, thin-QR. Similarity Analysis. | | | x | x | <i>in development</i> |
| Mahout Interactive Shell | | | | | |
| Interactive REPL shell for Spark optimized Mahout DSL | | | x | | |
| Collaborative Filtering | | | | | |
| User-Based Collaborative Filtering | x | | x | | |
| Item-Based Collaborative Filtering | x | x | x | | |
| Matrix Factorization with ALS | x | x | | | |
| Matrix Factorization with ALS on Implicit Feedback | x | x | | | |
| Weighted Matrix Factorization, SVD++ | x | | | | |
| Classification | | | | | |
| Logistic Regression - trained via SGD | x | | | | |
| Naive Bayes / Complementary Naive Bayes | | x | x | | |
| Random Forest | | x | | | |
| Hidden Markov Models | x | | | | |
| Multilayer Perceptron | x | | | | |

Mahout

Algoritmos

Es una buena librería para introducirse en el uso de MapReduce sobre Hadoop



| | Single Machine | MapReduce | Spark | H2O | Flink |
|--|-------------------|-------------------|-------|-----|-----------------------|
| Mahout Math-Scala Core Library and Scala DSL | | | | | |
| Mahout Distributed BLAS. Distributed Row Matrix API with R and Matlab like operators. Distributed ALS, SPCA, SSVD, thin-QR. Similarity Analysis. | | | x | x | <i>in development</i> |
| Clustering | | | | | |
| Canopy Clustering | <i>deprecated</i> | <i>deprecated</i> | | | |
| k-Means Clustering | x | x | | | |
| Fuzzy k-Means | x | x | | | |
| Streaming k-Means | x | x | | | |
| Spectral Clustering | | x | | | |
| Dimensionality Reduction <i>note: most scala-based dimensionality reduction algorithms are available through the Mahout Math-Scala Core Library for all engines</i> | | | | | |
| Singular Value Decomposition | x | x | x | x | |
| Lanczos Algorithm | <i>deprecated</i> | <i>deprecated</i> | | | |
| Stochastic SVD | x | x | x | x | |
| PCA (via Stochastic SVD) | x | x | x | x | |
| QR Decomposition | x | x | x | x | |

Spark Libraries



<https://spark.apache.org/>



[Download](#)

[Libraries ▾](#)

[Documentation ▾](#)

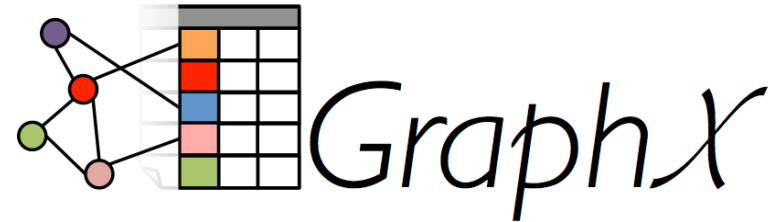
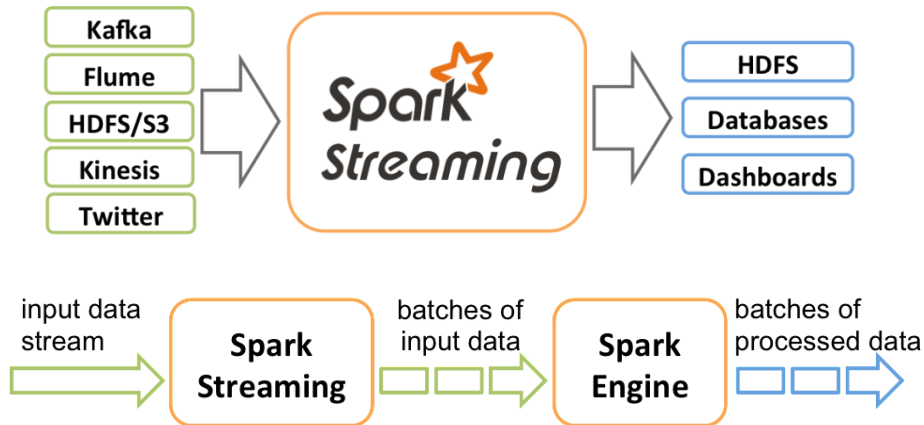
[Examples](#)

[Community ▾](#)

[FAQ](#)

- [APIs: RDD, DataFrame and SQL](#)
- [Backend Execution: DataFrame and SQL](#)
- [Integrations: Data Sources, Hive, Hadoop, Mesos and Cluster Management](#)
- [R Language](#)
- [Machine Learning and Advanced Analytics](#)
- [Spark Streaming](#)
- [Deprecations, Removals, Configs, and Behavior Changes](#)
 - [Spark Core](#)
 - [Spark SQL & DataFrames](#)
 - [Spark Streaming](#)
 - [MLlib](#)
- [Known Issues](#)
 - [SQL/DataFrame](#)
 - [Streaming](#)
- [Credits](#)

<https://spark.apache.org/docs/latest/mllib-guide.html>



MLlib



MLlib types, algorithms and utilities

This lists functionality included in `spark.mllib`, the main MLlib API.

- **Data types**
- **Basic statistics**
 - summary statistics
 - correlations
 - stratified sampling
 - hypothesis testing
 - random data generation
- **Classification and regression**
 - linear models (SVMs, logistic regression, linear regression)
 - naive Bayes
 - decision trees
 - ensembles of trees (Random Forests and Gradient-Boosted Trees)
 - isotonic regression
- **Collaborative filtering**
 - alternating least squares (ALS)
- **Clustering**
 - k-means
 - Gaussian mixture
 - power iteration clustering (PIC)
 - latent Dirichlet allocation (LDA)
 - streaming k-means
- **Dimensionality reduction**
 - singular value decomposition (SVD)
 - principal component analysis (PCA)
- **Feature extraction and transformation**
- **Frequent pattern mining**
 - FP-growth
- **Optimization (developer)**
 - stochastic gradient descent
 - limited-memory BFGS (L-BFGS)
- **PMML model export**

<https://spark.apache.org/mllib/>

<http://spark.apache.org/mllib/>

MLlib is Apache Spark's scalable machine learning library.

Ease of Use

Usable in Java, Scala and Python.

MLlib fits into Spark's APIs and interoperates with NumPy in Python (starting in Spark 0.9). You can use any Hadoop data source (e.g. HDFS, HBase, or local files), making it easy to plug into Hadoop workflows.

Performance

High-quality algorithms, 100x faster than MapReduce.

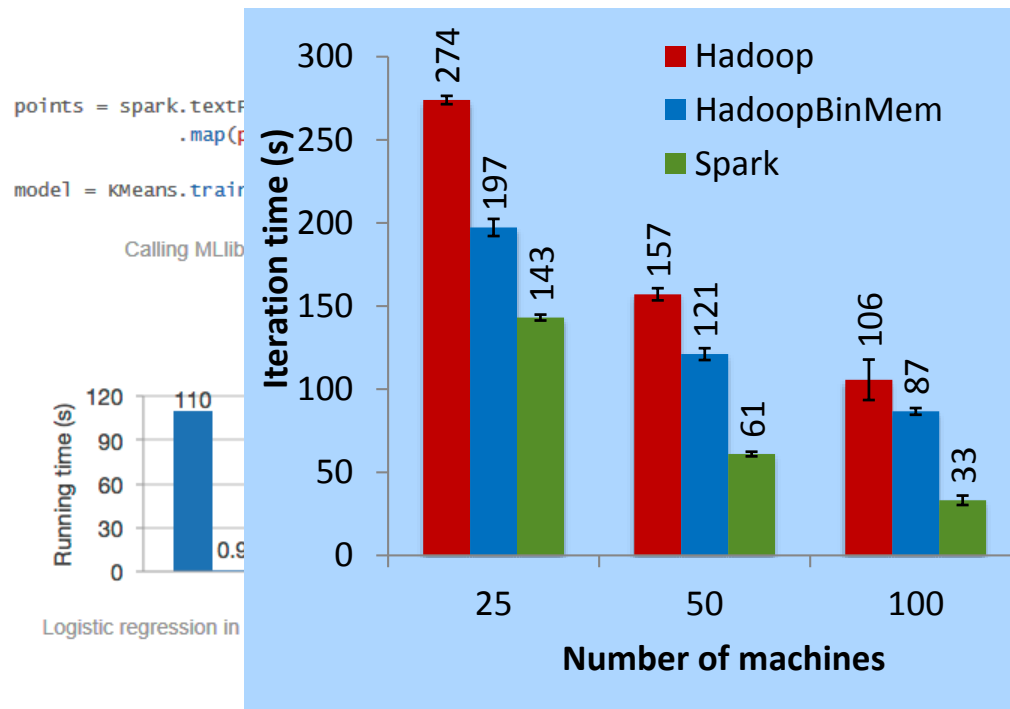
Spark excels at iterative computation, enabling MLlib to run fast. At the same time, we care about algorithmic performance: MLlib contains high-quality algorithms that leverage iteration, and can yield better results than the one-pass approximations sometimes used on MapReduce.

Easy to Deploy

Runs on existing Hadoop clusters and data.

If you have a Hadoop 2 cluster, you can run Spark and MLlib without any pre-installation. Otherwise, Spark is easy to run standalone or on EC2 or Mesos. You can read from HDFS, HBase, or any Hadoop data source.

K-Means



[Zaharia et. al, NSDI'12]



<https://spark.apache.org/docs/latest/mllib-guide.html>

Machine Learning Library (MLlib) Guide

MLlib is Spark's scalable machine learning library consisting of common learning algorithms and utilities, including classification, regression, clustering, collaborative filtering, dimensionality reduction, as well as underlying optimization primitives, as outlined below:

- [Data types](#)
- [Basic statistics](#)
 - summary statistics
 - correlations
 - stratified sampling
 - hypothesis testing
 - random data generation
- [Classification and regression](#)
 - linear models (SVMs, logistic regression, linear regression)
 - naive Bayes
 - decision trees
 - ensembles of trees (Random Forests and Gradient-Boosted Trees)
 - isotonic regression
- [Collaborative filtering](#)
 - alternating least squares (ALS)
- [Clustering](#)
 - k-means
 - Gaussian mixture
 - power iteration clustering (PIC)
 - latent Dirichlet allocation (LDA)
 - streaming k-means
- [Dimensionality reduction](#)
 - singular value decomposition (SVD)
 - principal component analysis (PCA)
- [Feature extraction and transformation](#)
- [Frequent pattern mining](#)
 - FP-growth
- [Optimization \(developer\)](#)
 - stochastic gradient descent
 - limited-memory BFGS (L-BFGS)

<http://spark-packages.org/>



Spark Packages

A community index of packages
for Apache Spark.

Data Science in H₂O

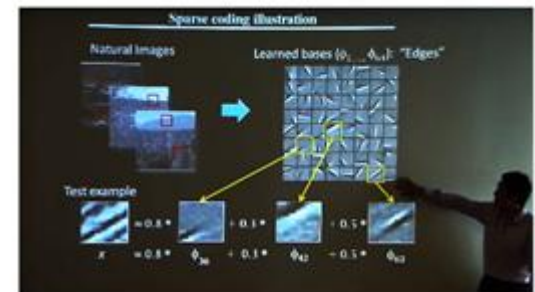
- Cox Proportional Hazards Model
- Deep Learning
- Generalized Linear Model
- Gradient Boosted Regression and Classification
- K-Means
- Naive Bayes
- Principal Components Analysis
- Random Forest
- Summary
- Data Science and Machine Learning
- Stochastic Gradient Descent
- References

■ Librería que contiene algoritmos de Deep Learning

- Récord del mundo en el problema MNIST sin preprocesamiento

Soporte para R, Hadoop y Spark

Funcionamiento: Crea una máquina virtual con Java en la que optimiza el paralelismo de los algoritmos



Deep learning
make sense of images, audio

H₂O APIs

<http://www.h2o.ai/resources/>

Overview and walkthroughs for the different APIs to H₂O.

- R On H₂O
- Tableau on H₂O



http://h2o-release.s3.amazonaws.com/h2o/rel-turan/4/docs-website/h2o-r/h2o_package.pdf

Machine Learning with Sparkling Water: H₂O + Spark



Sparkling Water allows users to combine the fast, scalable machine learning algorithms of H₂O with the capabilities of Spark. With Sparkling Water, users can drive computation from Scala/R/Python and utilize the H₂O Flow UI, providing an ideal machine learning platform for application developers.

Mahout. Caso de estudio

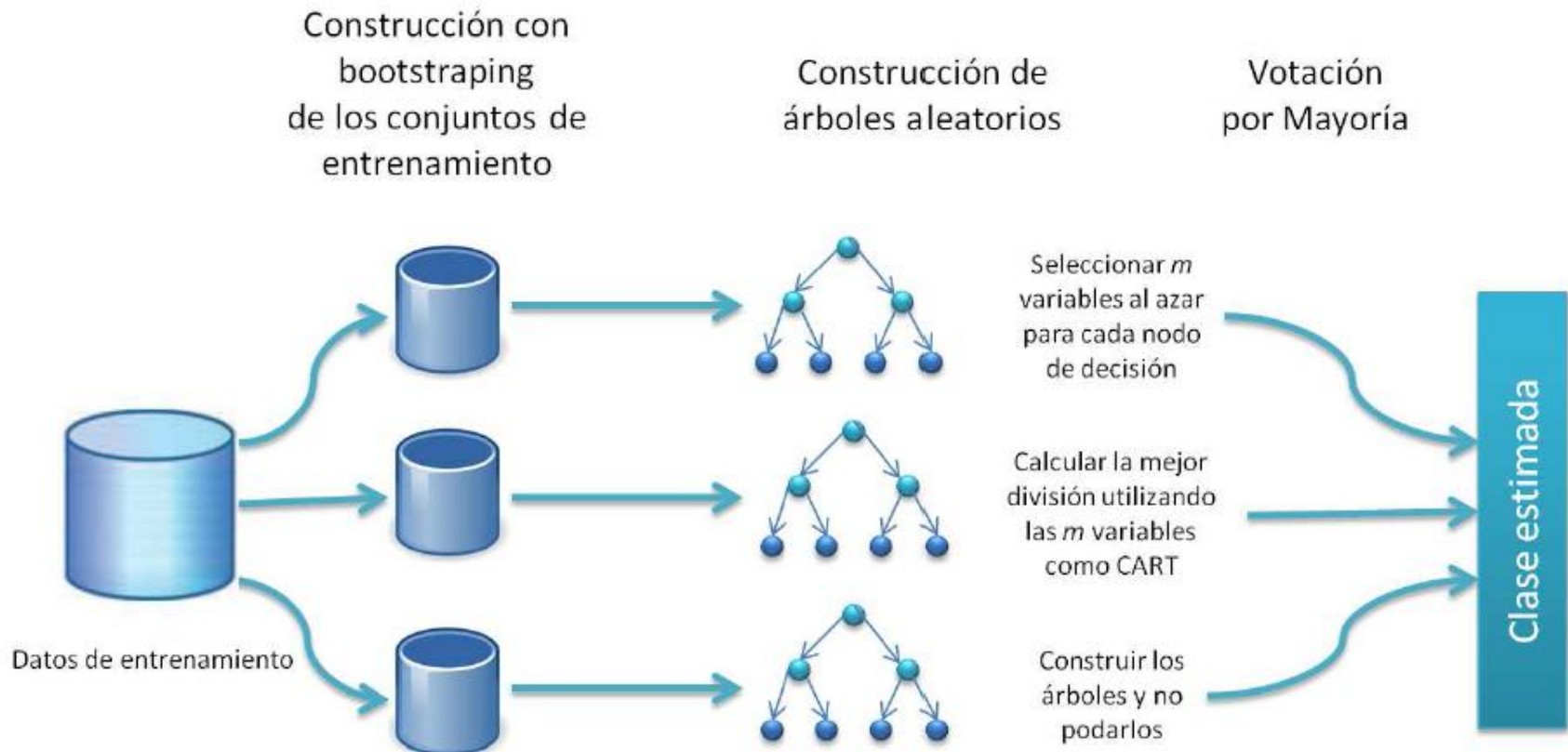


Scalable machine learning
and data mining



Apache Mahout has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining

Caso de estudio: Random Forest para KddCup99



Mahout. Caso de estudio



Scalable machine learning
and data mining



Apache Mahout has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining

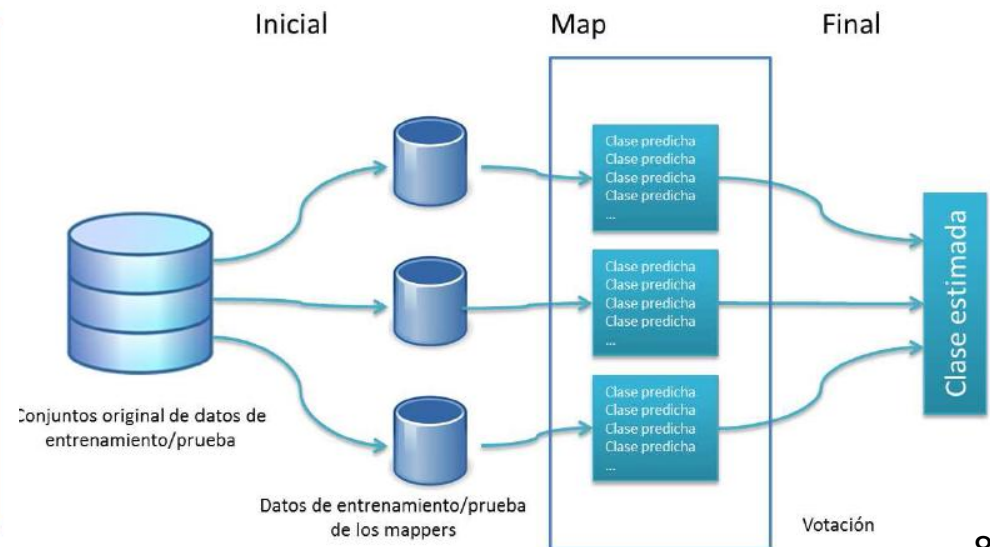
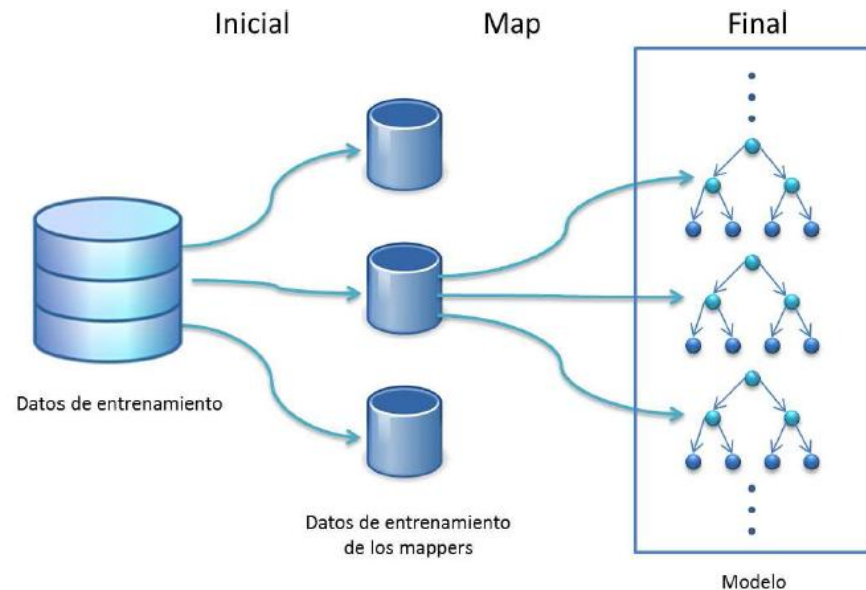
Caso de estudio: Random Forest para KddCup99

Implementación RF Mahout Partial: Es un algoritmo que genera varios árboles de diferentes partes de los datos (maps).

Dos fases:

Fase de Construcción

Fase de Clasificación



Mahout. Caso de estudio



Scalable machine learning
and data mining



Apache Mahout has implementations of a wide range of
machine learning and data mining algorithms:
clustering, classification, collaborative filtering and
frequent pattern mining

Caso de estudio: Random Forest para KddCup99

Tiempo en segundos para ejecución secuencial

| Class | Instance Number |
|--------|-----------------|
| normal | 972.781 |
| DOS | 3.883.370 |
| PRB | 41.102 |
| R2L | 1.126 |
| U2R | 52 |

| Datasets | RF | | |
|-------------------|---------|----------|----------|
| | 10% | 50% | full |
| DOS_versus_normal | 6344.42 | 49134.78 | NC |
| DOS_versus_PRB | 4825.48 | 28819.03 | NC |
| DOS_versus_R2L | 4454.58 | 28073.79 | NC |
| DOS_versus_U2R | 3848.97 | 24774.03 | NC |
| normal_versus_PRB | 468.75 | 6011.70 | NC |
| normal_versus_R2L | 364.66 | 4773.09 | 14703.55 |
| normal_versus_U2R | 295.64 | 4785.66 | 14635.36 |

Cluster ATLAS: 16 nodos

- Microprocessors: 2 x Intel E5-2620
(6 cores/12 threads, 2 GHz)
- RAM 64 GB DDR3 ECC 1600MHz
- Mahout version 0.8

Mahout. Caso de estudio



Scalable machine learning
and data mining



Apache Mahout has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining

Caso de estudio: Random Forest para KddCup99

| Class | Instance Number |
|--------|-----------------|
| normal | 972.781 |
| DOS | 3.883.370 |
| PRB | 41.102 |
| R2L | 1.126 |
| U2R | 52 |

| | 10% | 50% | full |
|-------------------|---------|----------|------|
| DOS_versus_normal | 6344.42 | 49134.78 | NC |
| DOS_versus_PRB | 4825.48 | 28819.03 | NC |

Tiempo en segundos para Big Data con 20 particiones

| Datasets | RF-BigData | | |
|-------------------|------------|-----|------|
| | 10% | 50% | full |
| DOS_versus_normal | 98 | 221 | 236 |
| DOS_versus_PRB | 100 | 186 | 190 |
| DOS_versus_R2L | 97 | 157 | 136 |
| DOS_versus_U2R | 93 | 134 | 122 |
| normal_versus_PRB | 94 | 58 | 72 |
| normal_versus_R2L | 92 | 39 | 69 |
| normal_versus_U2R | 93 | 52 | 64 |

Cluster ATLAS: 16 nodos
-Spark Random Forest: 43.50
seconds (20 partitions)

Aprendizaje no supervisado

Mahout

| Clustering | Single Machine | MapReduce |
|---------------------|-------------------|-------------------|
| Canopy Clustering | <i>deprecated</i> | <i>deprecated</i> |
| k-Means Clustering | x | x |
| Fuzzy k-Means | x | x |
| Streaming k-Means | x | x |
| Spectral Clustering | | x |

MLlib

- Frequent pattern mining
 - FP-growth
- Clustering
 - k-means
 - Gaussian mixture
 - power iteration clustering (PIC)
 - latent Dirichlet allocation (LDA)
 - streaming k-means

K-means

- **Mahout: The K-Means algorithm**
- **Input**
 - Dataset (set of points in 2D) –Large
 - Initial centroids (K points) –Small
- **Map Side**
 - Each map reads the K-centroids + one block from dataset
 - Assign each point to the closest centroid
 - Output <centroid, point>

R. M. Esteves, C. Rong, R. Pais, **K-means Clustering in the Cloud – A Mahout Test**.
IEEE Workshops of International Conference on Advanced Information Networking and
Applications, pp.514,519, 22-25 March 2011.

K-means

Mahout: K-means clustering

■ **Reduce Side**

- Gets all points for a given centroid
- Re-compute a new centroid for this cluster
- Output: <new centroid>

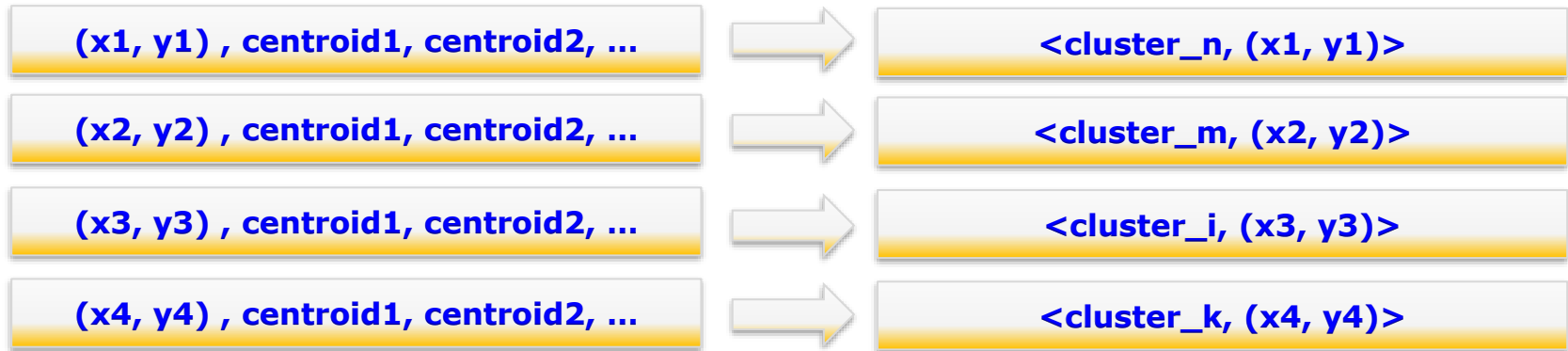
■ **Iteration Control**

- Compare the old and new set of K-centroids If similar or max iterations reached then Stop Else Start another Map-Reduce Iteration

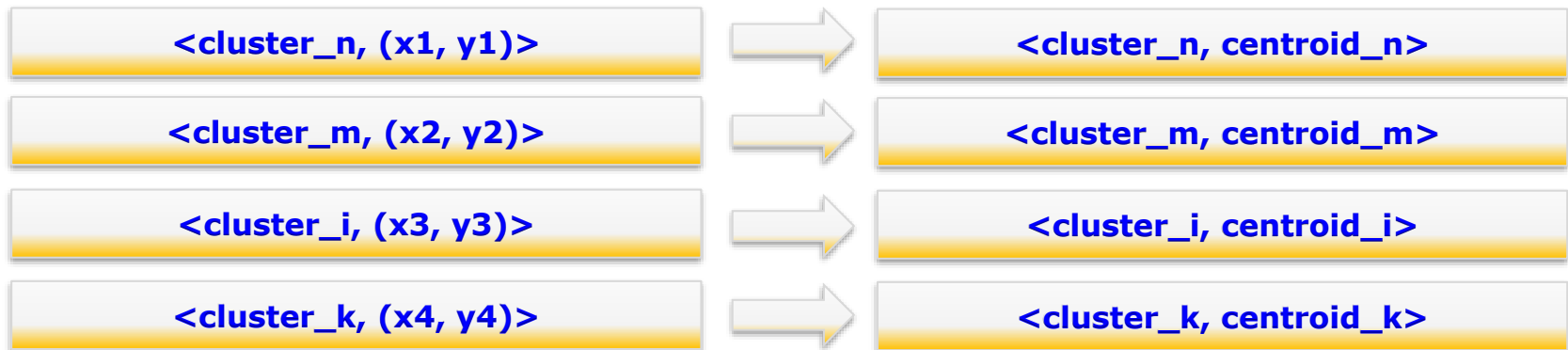
■ **THIS IS AN ITERATIVE MAP-REDUCE ALGORITHM**

K-means

- **Map phase: assign cluster IDs**

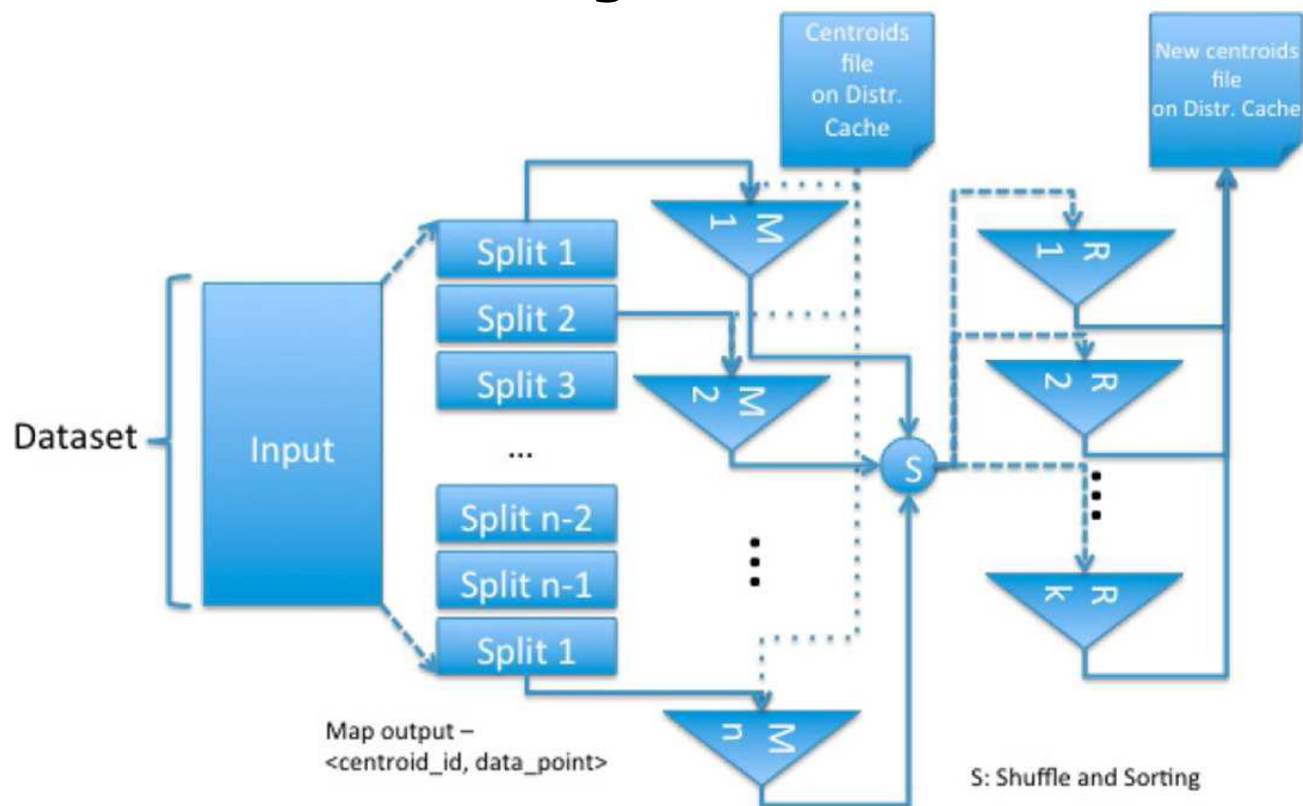


- **Reduce phase: reset centroids**



K-means

Mahout: K-means clustering



R. M. Esteves, C. Rong, R. Pais, **K-means Clustering in the Cloud – A Mahout Test**. IEEE Workshops of International Conference on Advanced Information Networking and Applications, pp.514,519, 22-25 March 2011.

K-means

K-Means: An example of limitation of MapReduce

- **What's wrong with these iterative approaches?**
 - Iterative algorithms in MapReduce chain multiple jobs together.
 - The standard MapReduce is not ready to do this.
 - Hadoop offers some snippets (Counters) to determine the stopping criteria.
- **Main issues:**
 - MapReduce jobs have high startup costs.
 - Repetitive Shuffle.
 - Results are serialized to HDFS.

K-means

K-Means Clustering using Spark

**Focus: Implementation
and Performance**

K-means

MLlib: K-means clustering

k-means pseudo-code:

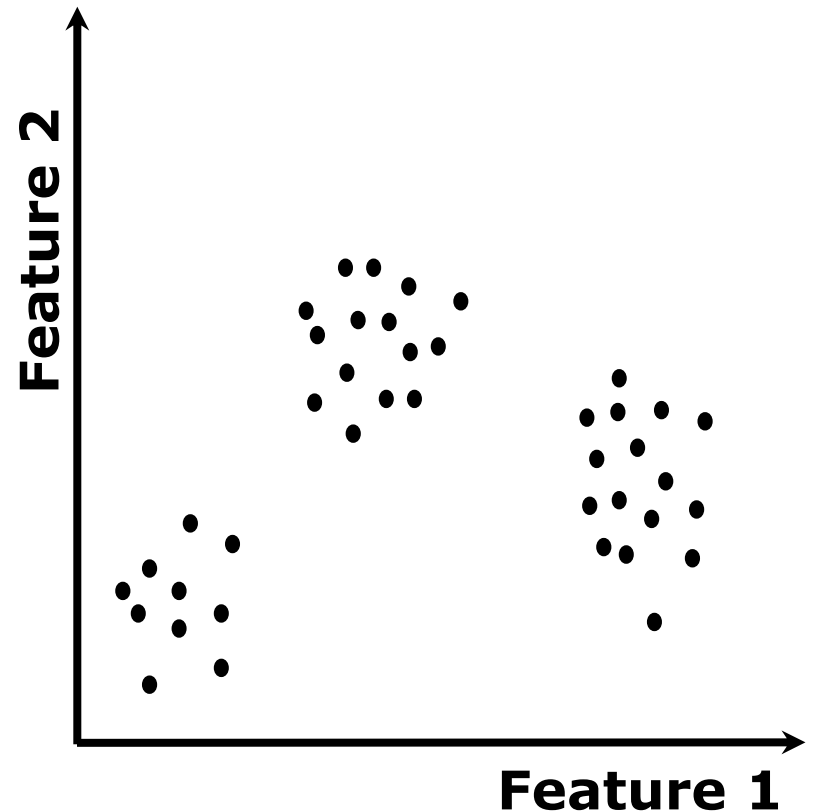
- Initialize K cluster centers
- Repeat until convergence:
 - Assign each data point to the cluster with the closest center.
 - Assign each cluster center to be the mean of its cluster's data points.

k-means MLlib source

```
centers = data.takeSample(
  false, K, seed)
while (d > ε)
{
  closest = data.map(p =>
    (closestPoint(p, centers), p))
  pointsGroup =
    closest.groupByKey()
  newCenters = pointsGroup.mapValues(
    ps => average(ps))
  d = distance(centers, newCenters)
  centers = newCenters.map(_)
```

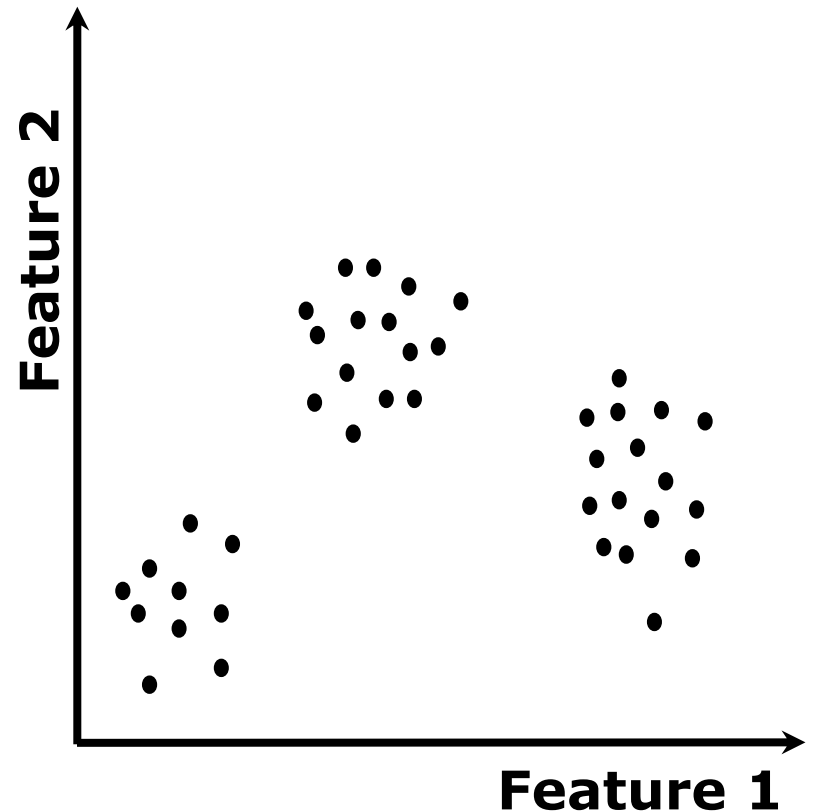
K-Means Algorithm

- Initialize K cluster centers
- Repeat until convergence:
 - Assign each data point to the cluster with the closest center.
 - Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

- **Initialize K cluster centers**
- Repeat until convergence:
 - Assign each data point to the cluster with the closest center.
 - Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

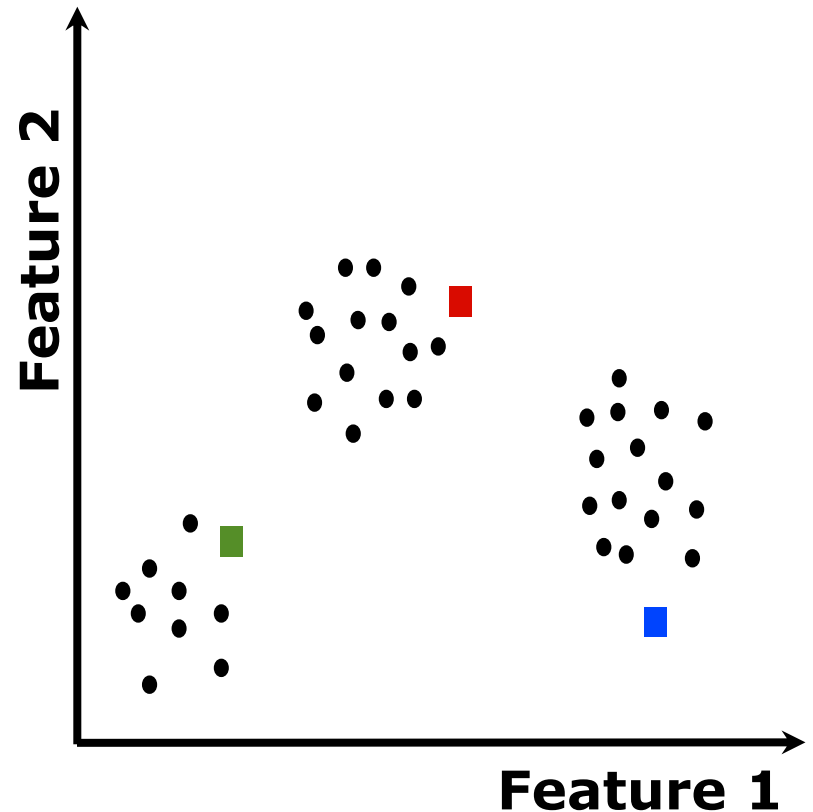
- **Initialize K cluster centers**

```
centers = data.takeSample(  
    false, K, seed)
```

- Repeat until convergence:

Assign each data point to the cluster with the closest center.

Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

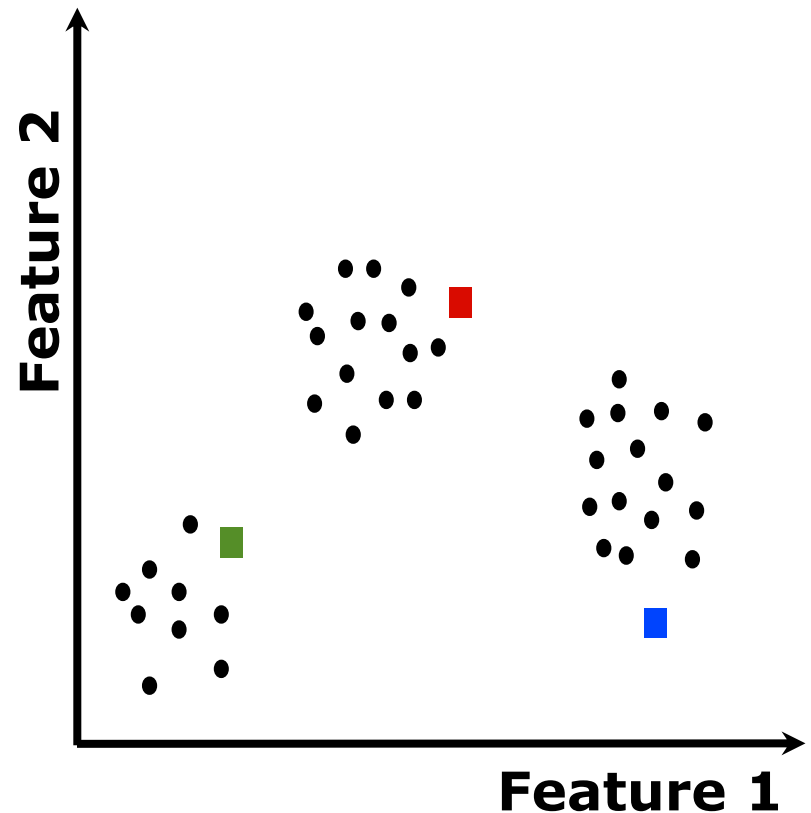
- Initialize K cluster centers

```
centers = data.takeSample(  
    false, K, seed)
```

- Repeat until convergence:

Assign each data point to the cluster with the closest center.

Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

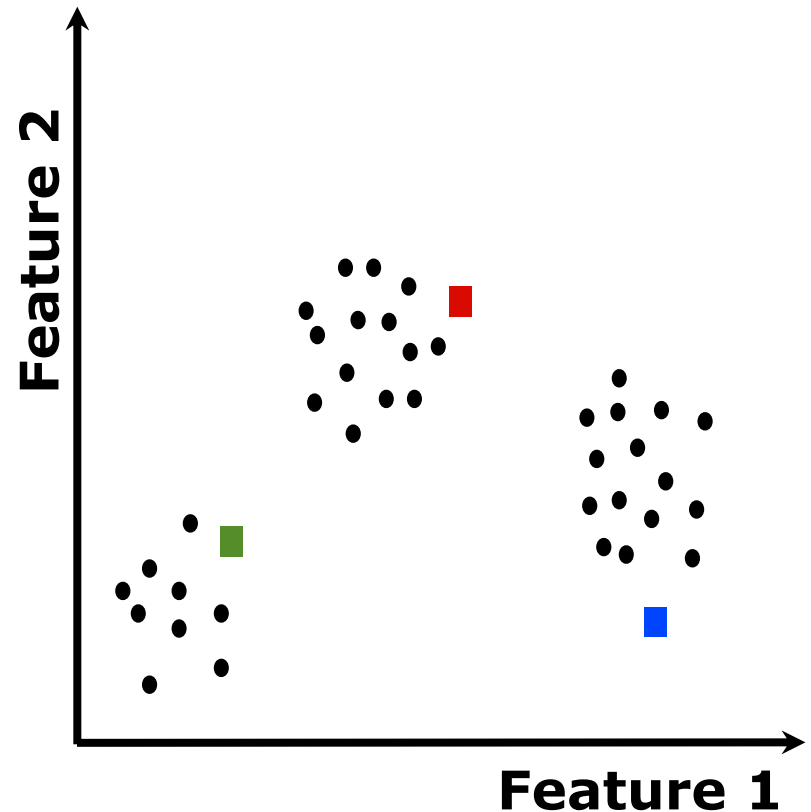
- Initialize K cluster centers

```
centers = data.takeSample(  
    false, K, seed)
```

- Repeat until convergence:

Assign each data point to the cluster with the closest center.

Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

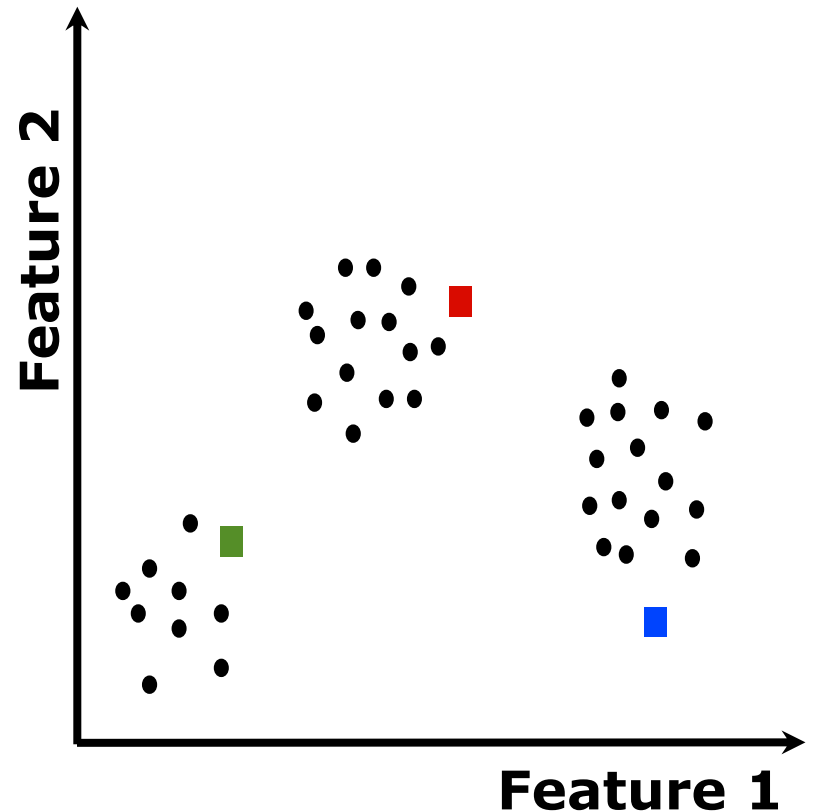
- Initialize K cluster centers

```
centers = data.takeSample(
  false, K, seed)
```

- Repeat until convergence:

```
closest = data.map(p =>
  (closestPoint(p, centers), p))
```

Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

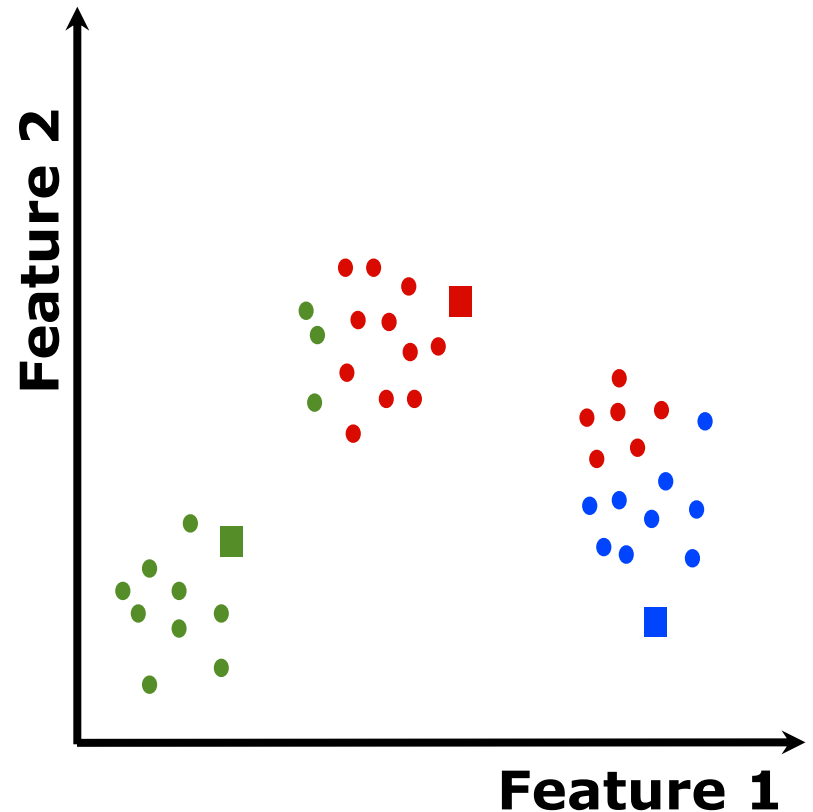
- Initialize K cluster centers

```
centers = data.takeSample(
  false, K, seed)
```

- Repeat until convergence:

```
closest = data.map(p =>
  (closestPoint(p, centers), p))
```

Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

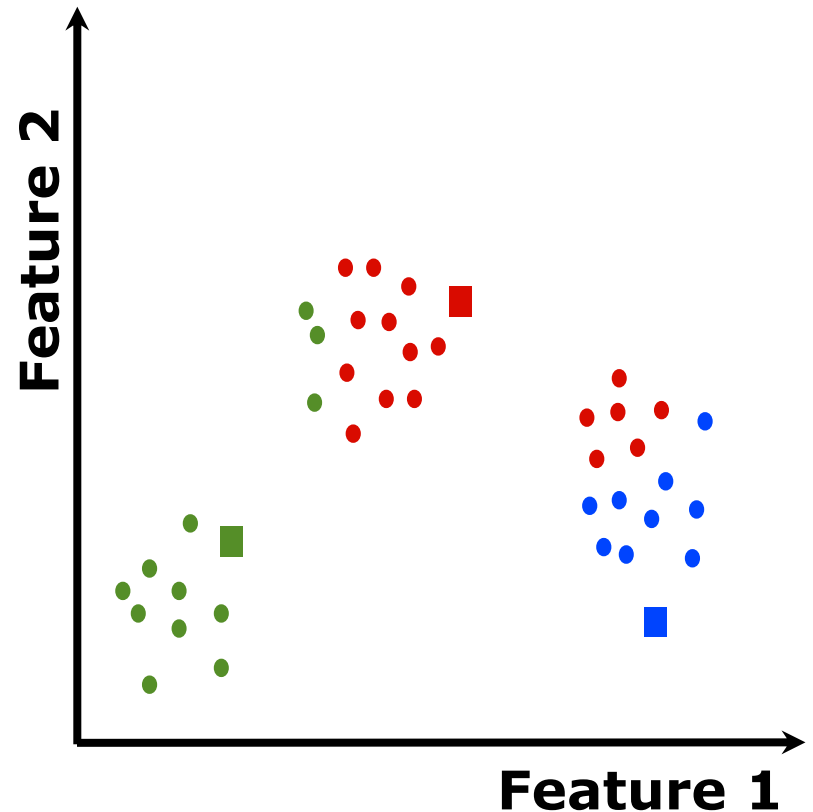
- Initialize K cluster centers

```
centers = data.takeSample(
  false, K, seed)
```

- Repeat until convergence:

```
closest = data.map(p =>
  (closestPoint(p, centers), p))
```

Assign each cluster center to be the mean of its cluster's data points.



K-Means Algorithm

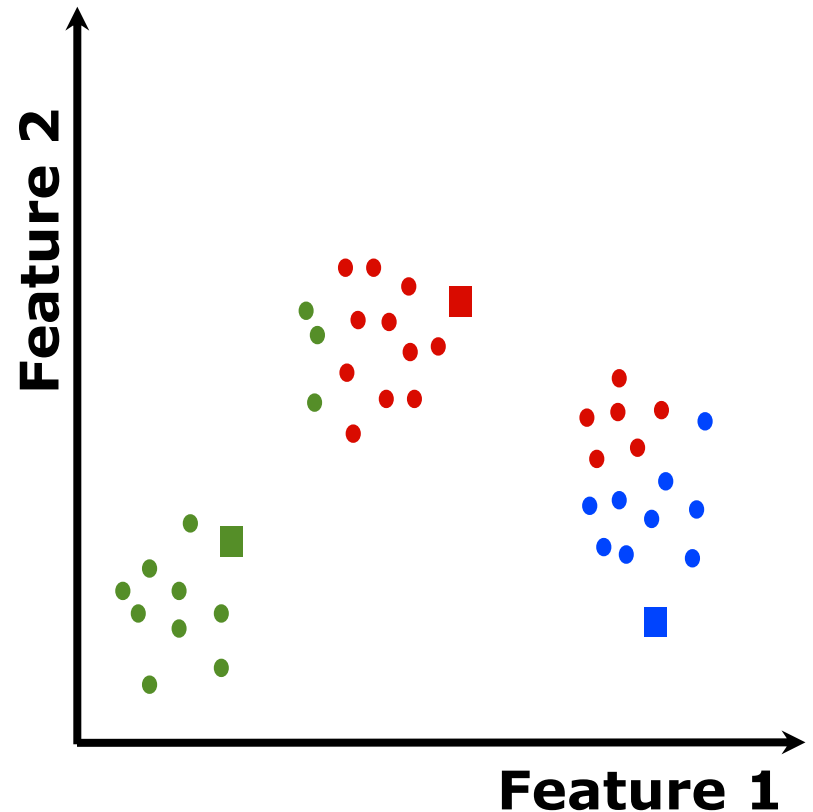
- Initialize K cluster centers

```
centers = data.takeSample(  
  false, K, seed)
```

- Repeat until convergence:

```
closest = data.map(p =>  
  
  (closestPoint(p, centers), p))
```

```
pointsGroup =  
  closest.groupByKey()
```



K-Means Algorithm

- Initialize K cluster centers

```
centers = data.takeSample(
  false, K, seed)
```

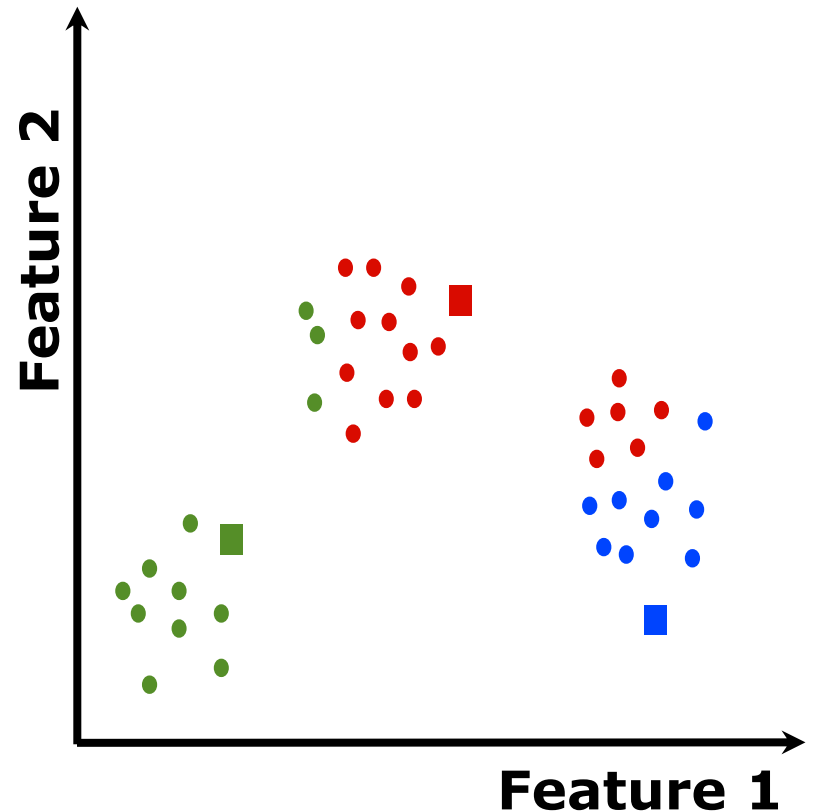
- Repeat until convergence:

```
closest = data.map(p =>
```

```
(closestPoint(p, centers), p))
```

```
pointsGroup =
  closest.groupByKey()
```

```
newCenters =
  pointsGroup.mapValues(
    ps => average(ps))
```



K-Means Algorithm

- Initialize K cluster centers

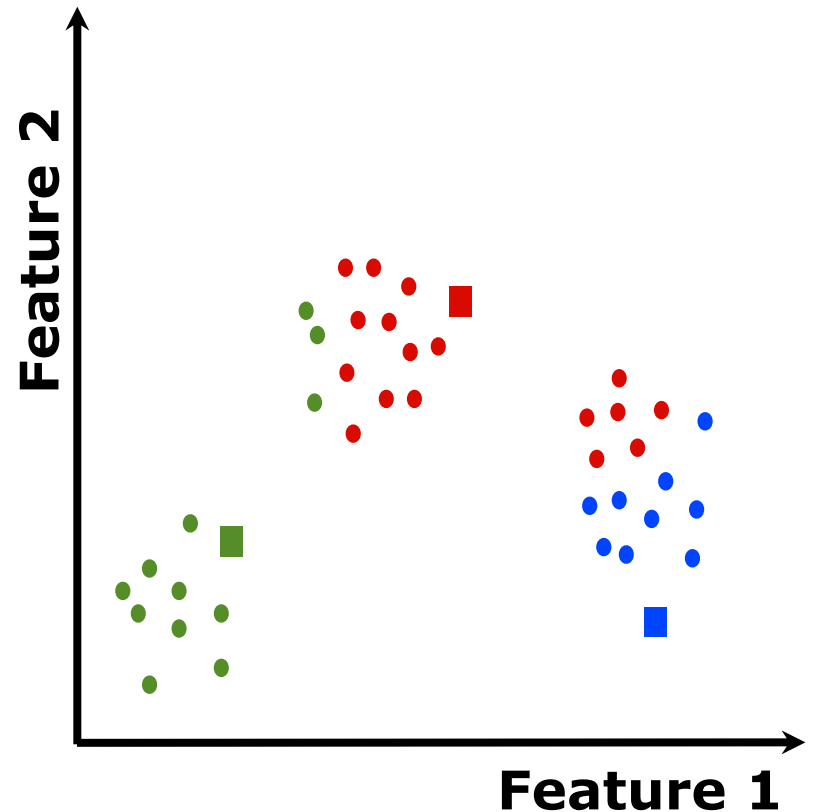
```
centers = data.takeSample(
  false, K, seed)
```

- Repeat until convergence:

```
closest = data.map(p =>
  (closestPoint(p,centers),p))
```

```
pointsGroup =
  closest.groupByKey()
```

```
newCenters =
  pointsGroup.mapValues(
    ps => average(ps))
```



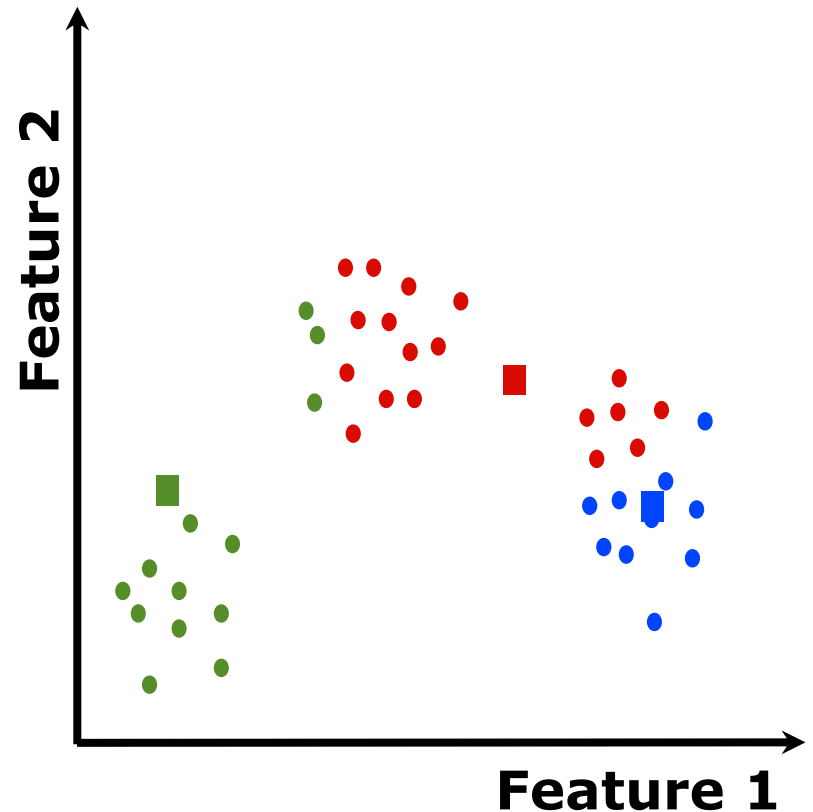
K-Means Algorithm (proceso iterativo utilizando los bloques en memoria)

- Initialize K cluster centers

```
centers = data.takeSample(
  false, K, seed)
```

- Repeat until convergence:

```
closest = data.map(p =>
  (closestPoint(p, centers), p))
pointsGroup =
  closest.groupByKey()
newCenters =
  pointsGroup.mapValues(
    ps => average(ps))
```



K-Means Algorithm

- Initialize K cluster centers

```
centers = data.takeSample(
  false, K, seed)
```

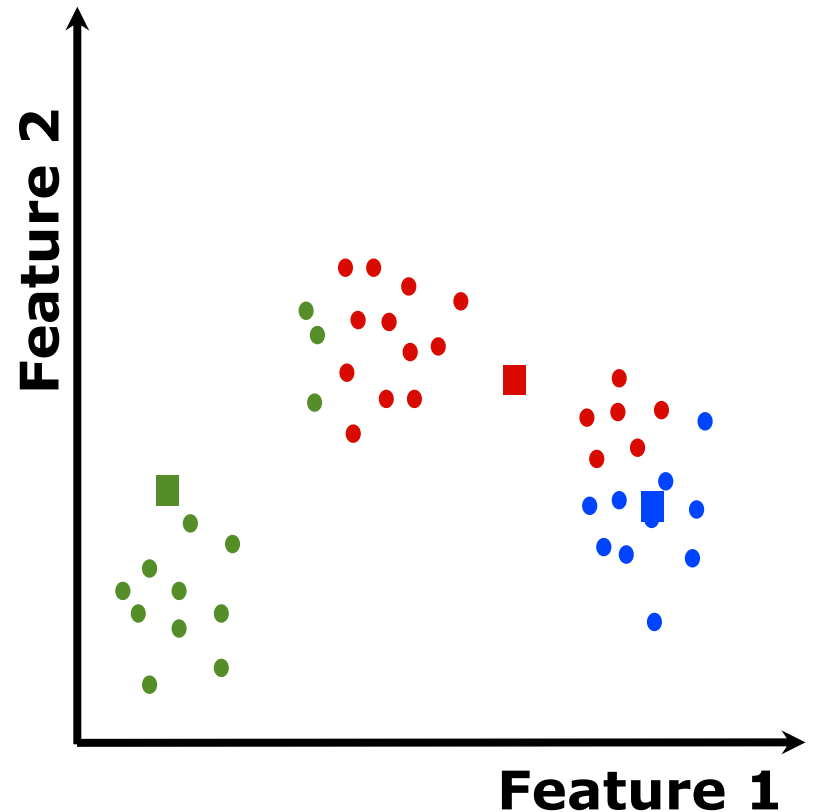
- Repeat until convergence:

```
while (dist(centers, newCenters)
  >  $\epsilon$ )
```

```
closest = data.map(p =>
  (closestPoint(p, centers), p))
```

```
pointsGroup =
  closest.groupByKey()
```

```
newCenters
=pointsGroup.mapValues(
  ps => average(ps))
```



K-Means Algorithm

- Initialize K cluster centers

```
centers = data.takeSample(
  false, K, seed)
```

- Repeat until convergence:

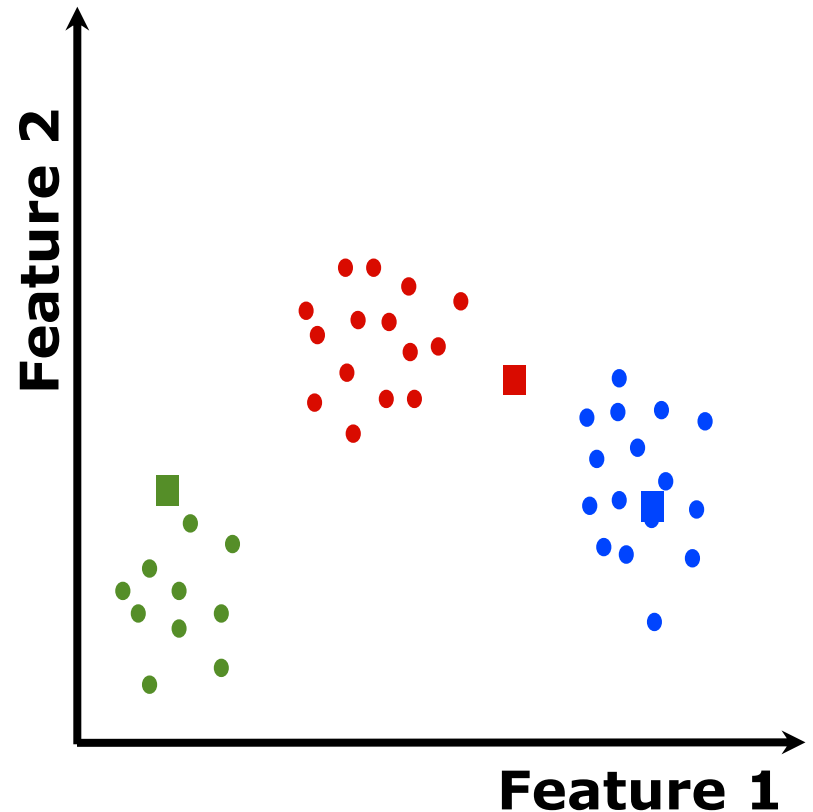
```
while (dist(centers, newCenters)
  >  $\epsilon$ )
```

```
closest = data.map(p =>
```

```
(closestPoint(p, centers), p))
```

```
pointsGroup =
  closest.groupByKey()
```

```
newCenters
=pointsGroup.mapValues(
  ps => average(ps))
```



K-Means Source

```
centers = data.takeSample(
  false, K, seed)

while (d > ε)
{
  closest = data.map(p =>

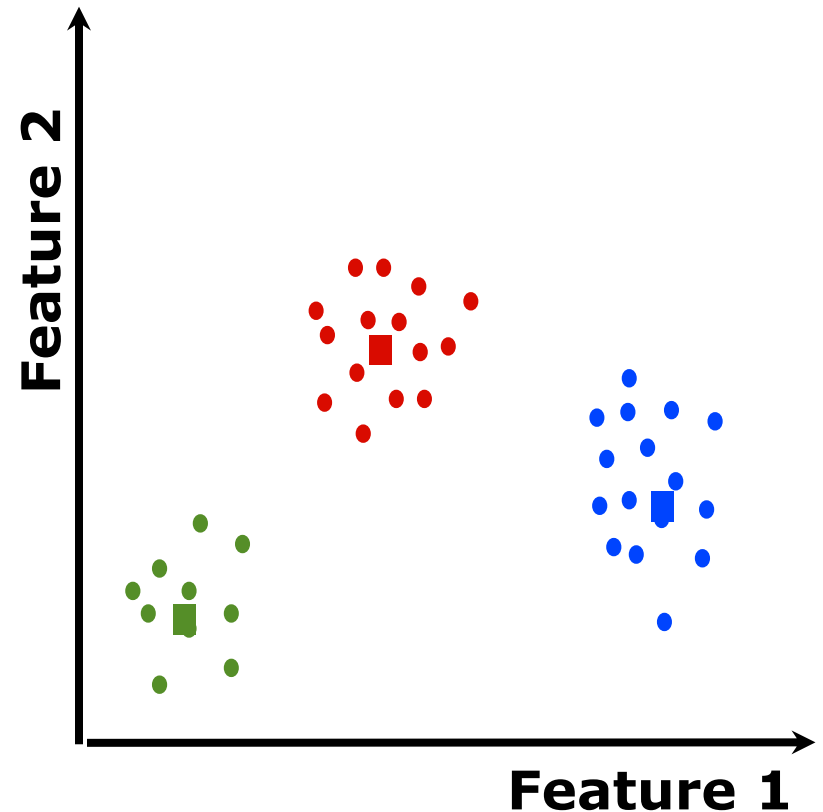
    (closestPoint(p,centers),p))

  pointsGroup =
    closest.groupByKey()

  d = distance(centers, newCenters)

  newCenters
  =pointsGroup.mapValues(
    ps => average(ps))

  centers = newCenters.map(_)
}
```

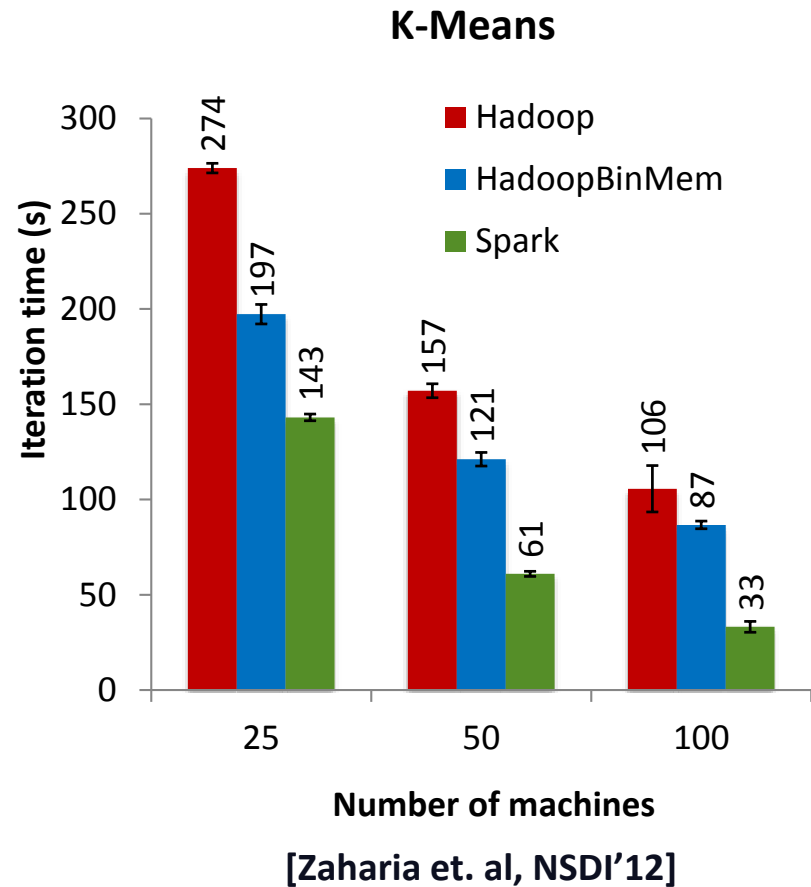


K-Means Performance

Lines of code for K-Means

Spark ~ 90 lines –

Hadoop ~ 4 files, > 300 lines



Aprendizaje no supervisado

Mlib: FP-growth

Mlib implements a parallel version of FP-growth called PFP, as described in

Li et al., PFP: Parallel FP-growth for query recommendation. RecSys'08 Proceedings of the 2008 ACM conference on Recommender systems Pages 107-114

PFP distributes the work of growing FP-trees based on the suffices of transactions, and hence more scalable than a single-machine implementation.

Big Data Analytics: **2** Comentarios finales

Image Credit: [Shutterstock](#)



**Without Analytics,
Big Data is Just
Noise**

*Guest post by Eric
Schwartzman, founder and
CEO of [Comply Socially](#)*

Big Data Analytics: 2 Comentarios finales

Mahout



| Classification | Single Machine | MapReduce |
|---|----------------|-----------|
| Logistic Regression - trained via SGD | X | |
| Naive Bayes / Complementary Naive Bayes | | X |
| Random Forest | | X |
| Hidden Markov Models | X | |
| Multilayer Perceptron | X | |

Casi ausencia de algoritmos para big data preprocessing

MLlib types, algorithms and utilities

This lists functionality included in spark.mllib, the main MLlib API.

- Data types
- Basic statistics
 - summary statistics
 - correlations
 - stratified sampling
 - hypothesis testing
 - random data generation
- Classification and regression
 - linear models (SVMs, logistic regression, linear regression)
 - naive Bayes
 - decision trees
 - ensembles of trees (Random Forests and Gradient-Boosted Trees)
 - isotonic regression
- Collaborative filtering
 - alternating least squares (ALS)

- Clustering
 - k-means
 - Gaussian mixture
 - power iteration clustering (PIC)
 - latent Dirichlet allocation (LDA)
 - streaming k-means
- Dimensionality reduction
 - singular value decomposition (SVD)
 - principal component analysis (PCA)
- Feature extraction and transformation
- Frequent pattern mining
 - FP-growth
- Optimization (developer)
 - stochastic gradient descent
 - limited-memory BFGS (L-BFGS)
- PMML model export

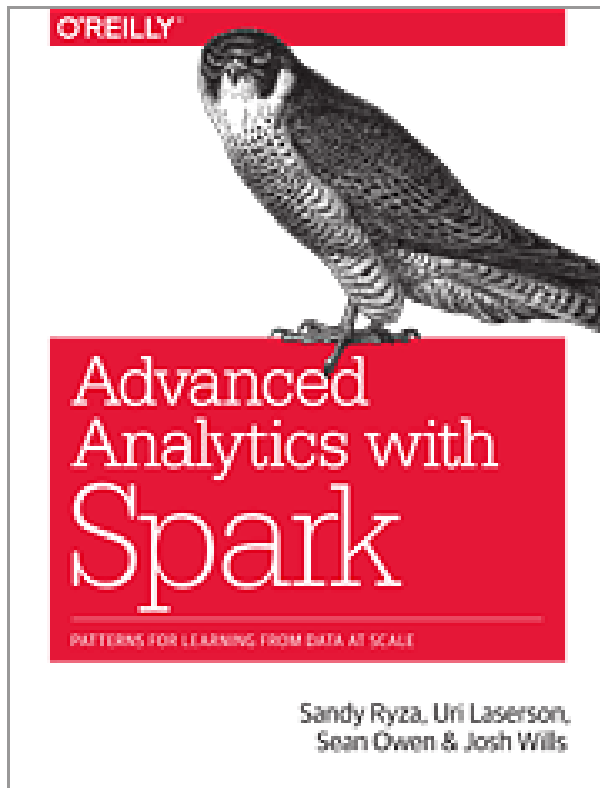


MLlib

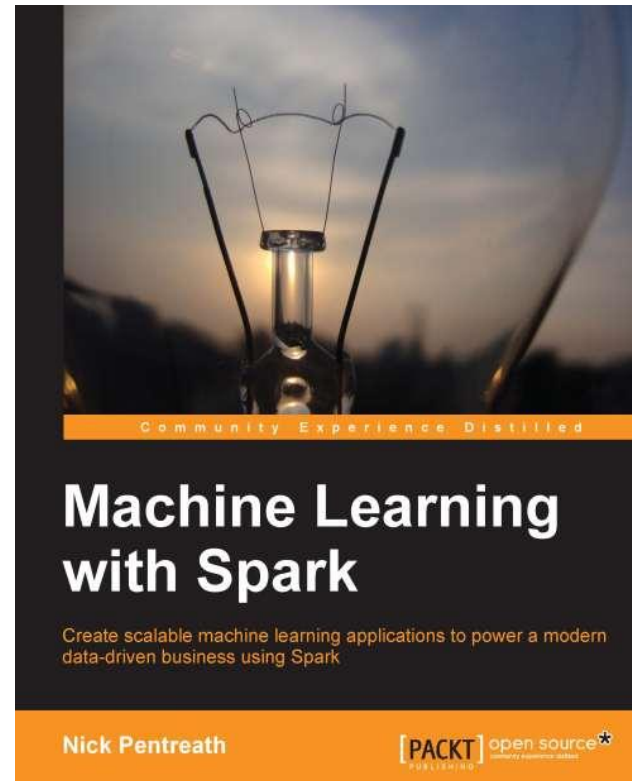


<https://spark.apache.org/mllib/>
Version 1.4.1

Big Data Analytics: 2 libros

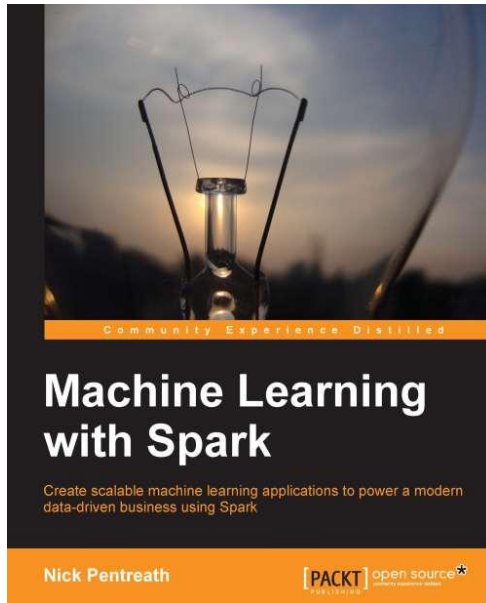


9 cases of study



10 chapters giving a quick glance on Machine Learning with Spark

Big Data Preprocessing



A short introduction to data preparation with Spark – Chapter 3

Chapter 3: Obtaining, Processing, and Preparing Data with Spark

Accessing publicly available datasets

- The MovieLens 100k dataset

Exploring and visualizing your data

- Exploring the user dataset
- Exploring the movie dataset
- Exploring the rating dataset

Processing and transforming your data

- Filling in bad or missing data

Extracting useful features from your data

- Numerical features
- Categorical features
- Derived features
 - Transforming timestamps into categorical features
- Text features
 - Simple text feature extraction
- Normalizing features
 - Using MLlib for feature normalization
 - Using packages for feature extraction

Summary

Big Data Preprocessing

<https://spark.apache.org/docs/latest/mllib-guide.html>



MLlib - Feature Extraction and Transformation

- TF-IDF
- Word2Vec
 - Model
 - Example
- StandardScaler
 - Model Fitting
 - Example
- Normalizer
 - Example
- Feature selection
 - ChiSqSelector
 - Model Fitting
 - Example



MLlib - Dimensionality Reduction

- Singular value decomposition (SVD)
 - Performance
 - SVD Example
- Principal component analysis (PCA)

ChiSqSelector

ChiSqSelector stands for Chi-Squared feature selection. It operates on labeled data with categorical features. ChiSqSelector orders features based on a Chi-Squared test of independence from the class, and then filters (selects) the top features which are most closely related to the label.

Model Fitting

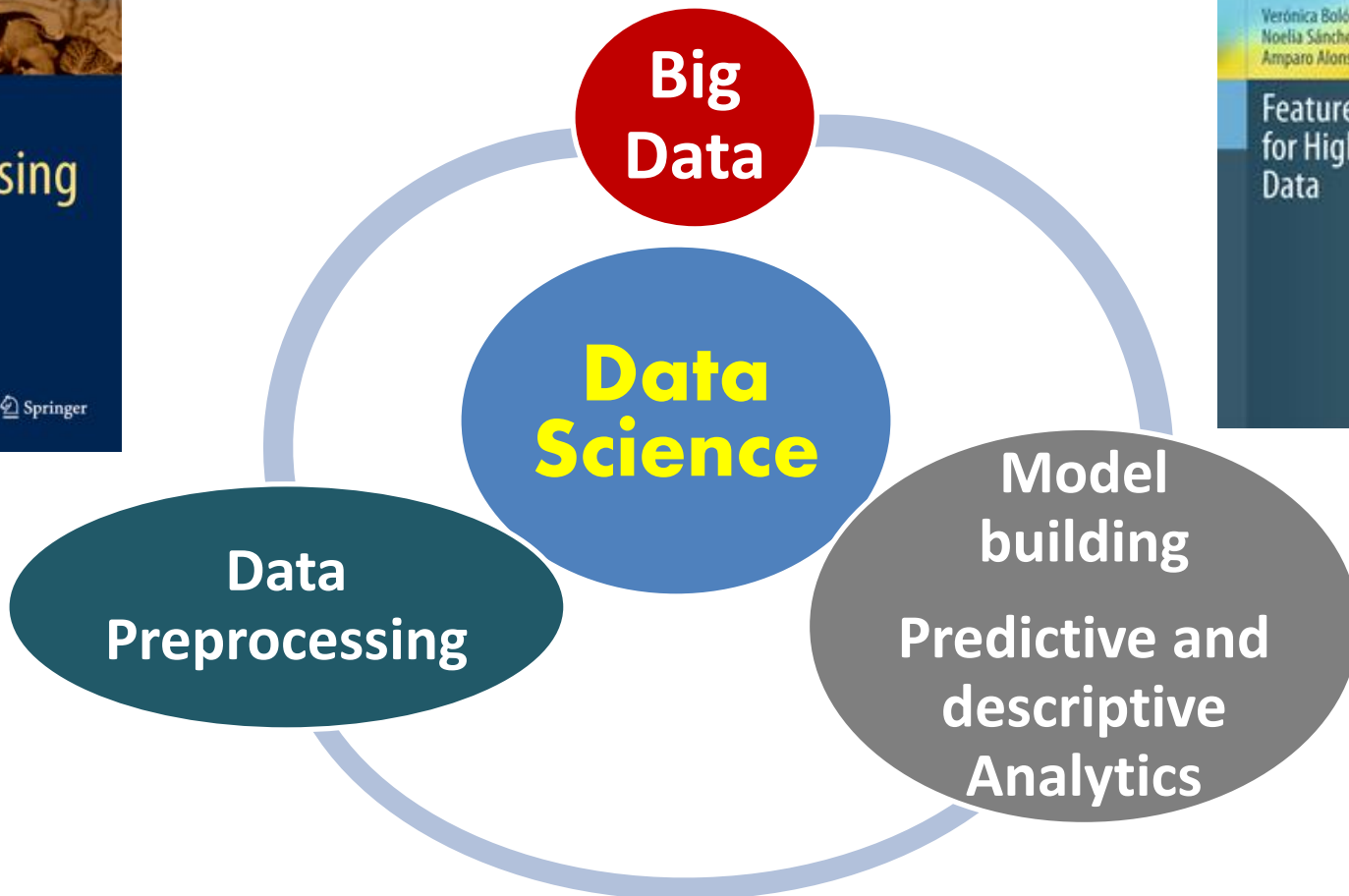
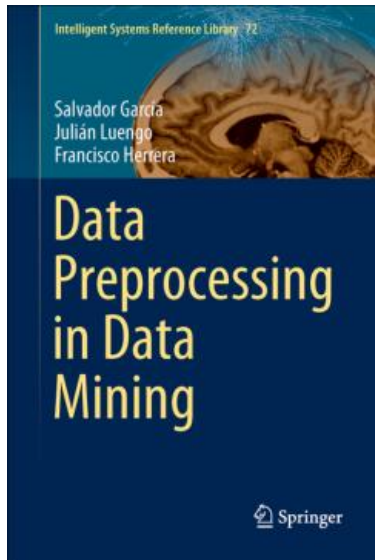
ChiSqSelector has the following parameter in the constructor:

- numTopFeatures number of top features that the selector will select (filter).

Big Data Analytics:

Big Data Preprocessing

¡Se requieren datos de calidad para diseñar modelos de calidad!.



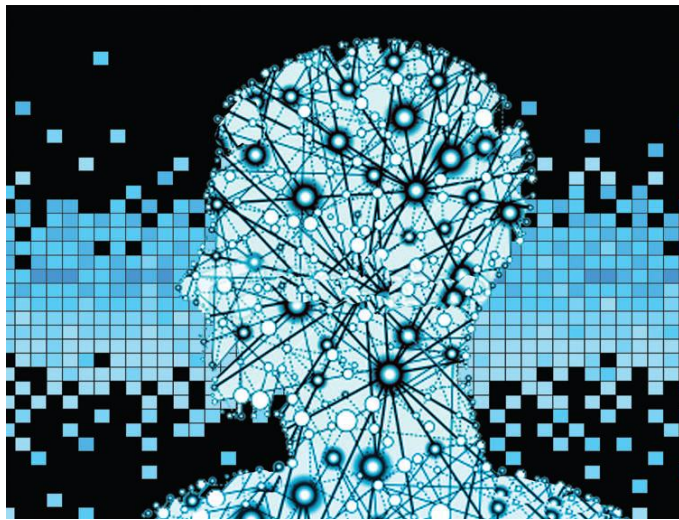
BIG DATA

Índice

- ❑ Big Data. Big Data Science
- ❑ ¿Por qué Big Data? Google crea el Modelo de Programación MapReduce
- ❑ Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...)
- ❑ Big Data Analytics: Librerías para Analítica de Datos en Big Data. Casos de estudio
- ❑ **Algunas aplicaciones: Salud, Social Media, Identificación**
- ❑ Comentarios Finales

Algunas aplicaciones: Salud

HEALTHCARE ORGANIZATIONS HAVE COLLECTED EXABYTES OF DATA LAST YEARS



BIG DATA AND DISEASE PREVENTION:

*From Quantified Self to Quantified
Communities*

Meredith A. Barrett,^{1,2} Olivier Humblet,^{1,2*}
Robert A. Hiatt,³ and Nancy E. Adler¹*

Big data can facilitate action on the modifiable risk factors that contribute to a large fraction of the chronic disease burden, such as physical activity, diet, tobacco use, and exposure to pollution.

Wikipedia y la detección de la gripe

Wikipedia is better than Google at tracking flu trends?

Detect pandemic risk in real time

Wikipedia traffic could be used to provide real time tracking of flu cases, according to the study published by John Brownstein.

Wikipedia Usage Estimates Prevalence of Influenza-Like Illness in the United States in Near Real-Time

David J. McIver , John S. Brownstein, Harvard Medical School, Boston Children's Hospital, Plos Computational Biology, 2014.

Redes sociales, big data y medidas de salud pública

Studies: Health, Social Media and Big Data

You Are What You Tweet: Analyzing Twitter for Public Health

Discovering Health Topics in Social Media Using Topic Models

Michael J. Paul, Mark Dredze, Johns Hopkins University, Plos One, 2014

Analyzing user messages in social media can measure different population characteristics, including public health measures.

A system filtering Twitter data can automatically infer health topics in 144 million Twitter messages from 2011 to 2013. ATAM discovered 13 coherent clusters of Twitter messages, some of which correlate with seasonal influenza ($r = 0.689$) and allergies ($r = 0.810$) temporal surveillance data, as well as exercise ($r = .534$) and obesity ($r = 2.631$) related geographic survey data in the United States.

Algunas aplicaciones



TV AUDIENCE MEASUREMENT WITH BIG DATA

Shawndra Hill

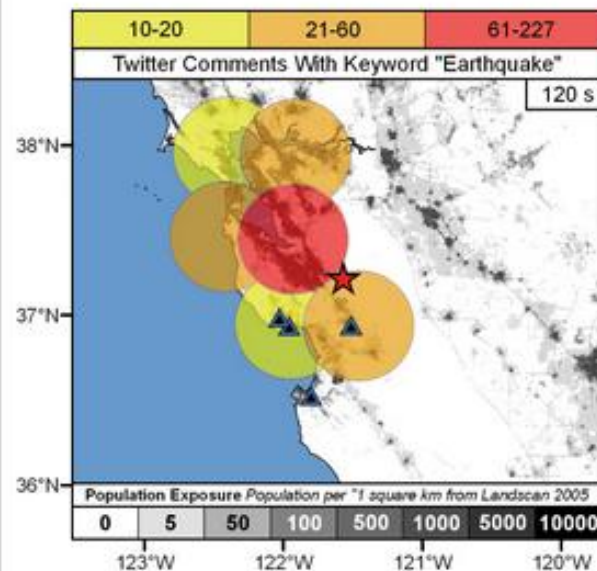
*Operations and Information Management,
University of Pennsylvania, Philadelphia, Pennsylvania*

“COMBINING MARKETING AND
INFORMATION SYSTEMS
METHODS, SOCIAL MEDIA
TEXT MINING, AND
TRADITIONAL CLV METHODS,
WE ARE ABLE TO PREDICT
FUTURE VIEWERSHIP ON AN
INDIVIDUAL LEVEL.”

Algunas aplicaciones

Earthquakes and Social Media

U.S. Geological Survey: Twitter Earthquake Detector (TED)



The United States Geological Survey Twitter searches increases in the volume of messages on earthquake and has been able to locate earthquakes with 90% accuracy.

Algunas aplicaciones: La banca es un ámbito de aplicación importante

Expansión.com

Domingo, 08.12.13. Actualizado a las 11:02

BBVA da un paso al frente en 'big data'

El banco ha actualizado todo su sistema de almacenamiento de datos corporativo para tener acceso desde un único punto a todas las 'islas de información' de la entidad en el mundo y consolidarse como un referente mundial en innovación tecnológica.



BBVA

Innova Challenge API

Available statistics services

Algunas aplicaciones



PRIVACIDAD EN INTERNET »

Cuatro compras con la tarjeta bastan para identificar a cualquier persona

- Los patrones de uso de las tarjetas permiten descubrir la identidad del 90% de una muestra de 1,1 millones de personas anónimas, según demuestra un estudio del MIT

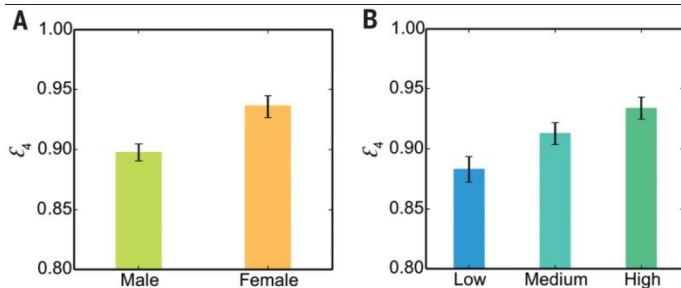
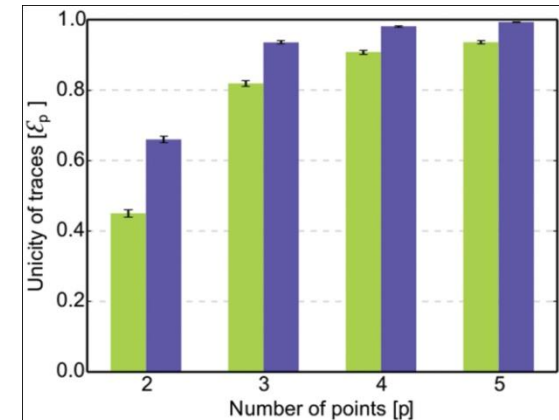


IDENTITY AND PRIVACY

Unique in the shopping mall: On the reidentifiability of credit card metadata

Yves-Alexandre de Montjoye,^{1*} Laura Radaelli,² Vivek Kumar Singh,^{1,3} Alex "Sandy" Pentland¹

Large-scale data sets of human behavior have the potential to fundamentally transform the way we fight diseases, design cities, or perform research. Metadata, however, contain sensitive information. Understanding the privacy of these data sets is key to their broad use and, ultimately, their impact. We study 3 months of credit card records for 1.1 million people and show that four spatiotemporal points are enough to uniquely reidentify 90% of individuals. We show that knowing the price of a transaction increases the risk of reidentification by 22%, on average. Finally, we show that even data sets that provide coarse information at any or all of the dimensions provide little anonymity and that women are more reidentifiable than men in credit card metadata.



<http://www.sciencemag.org/content/347/6221/536>

http://elpais.com/elpais/2015/01/29/ciencia/1422520042_066660.html

Algunas aplicaciones

Evolutionary Computation for Big Data and Big Learning Workshop

ECBDL'14 Big Data Competition 2014: Self-deployment track

Objective: Contact map prediction

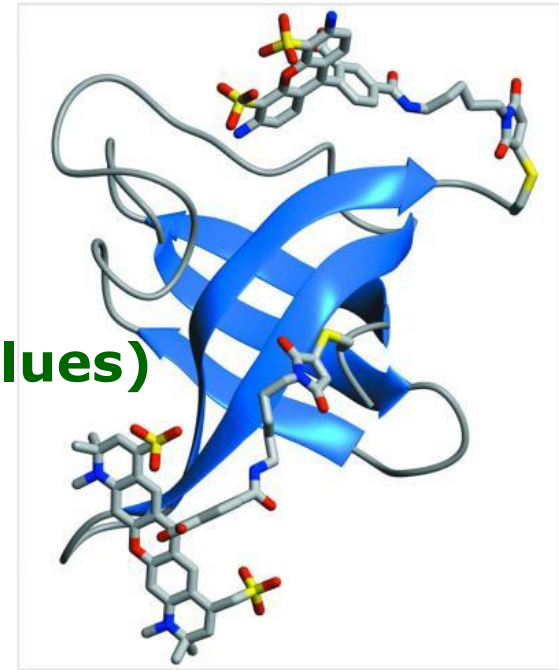
Details:

- ❑ 32 million instances
- ❑ 631 attributes (**539 real & 92 nominal values**)
- ❑ 2 classes
- ❑ 98% of negative examples
- ❑ About 56.7GB of disk space

Evaluation:

True positive rate · True negative rate
TPR · TNR

<http://cruncher.ncl.ac.uk/bdcomp/index.pl?action=data>



J. Bacardit et al, Contact map prediction using a large-scale ensemble of rule sets and the fusion of multiple predicted structural features, *Bioinformatics* 28 (19) (2012) 2441-2448

Evolutionary Computation for Big Data and Big Learning Workshop

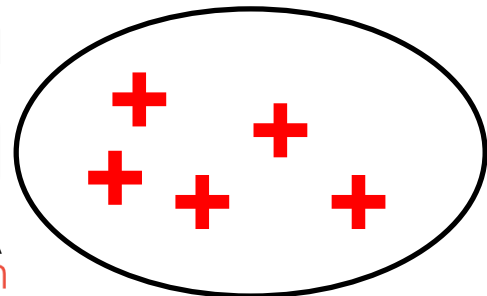
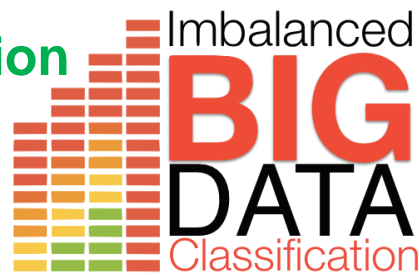
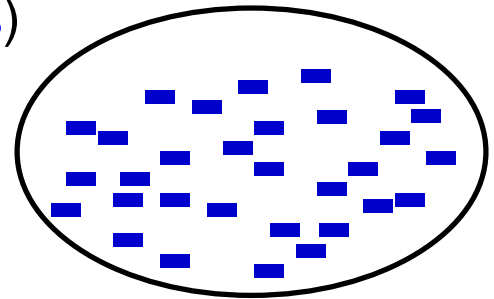
ECBDL'14 Big Data Competition 2014: Self-deployment track

The challenge:

- ❑ Very **large** size of the training set
 - ❑ Does not fit all together in memory.
- ❑ Even large for the **test set** (5.1GB, 2.9 million instances)
- ❑ Relatively **high dimensional** data.
- ❑ Low ratio (<2%) of true contacts. Imbalance rate: > 49

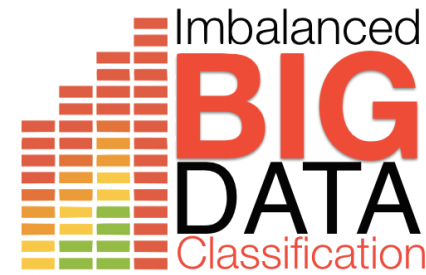
❑ **Imbalanced problem!**

❑ **Imbalanced Big Data Classification**



Imbalanced Big Data Classification

A MapReduce Approach



32 million instances, 98% of negative examples

Low ratio of true contacts (<2%). Imbalance rate: > 49.

Imbalanced problem!

Previous study on extremely imbalanced big data:

S. Río, V. López, J.M. Benítez, F. Herrera, On the use of MapReduce for Imbalanced Big Data using Random Forest. *Information Sciences* 285 (2014) 112-137.

Over-Sampling

Random

Focused

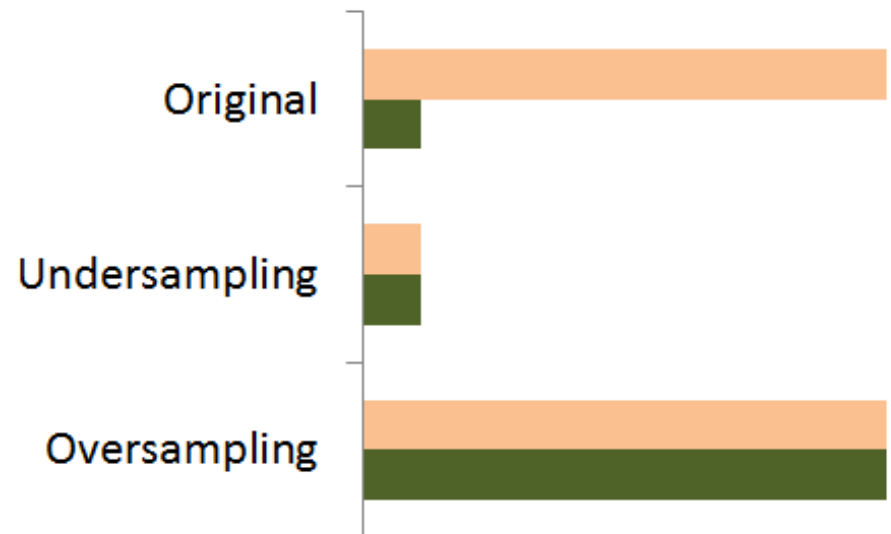
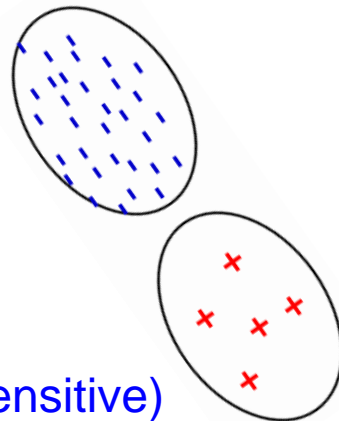
Under-Sampling

Random

Focused

Cost Modifying (cost-sensitive)

Boosting/Bagging approaches (with preprocessing)

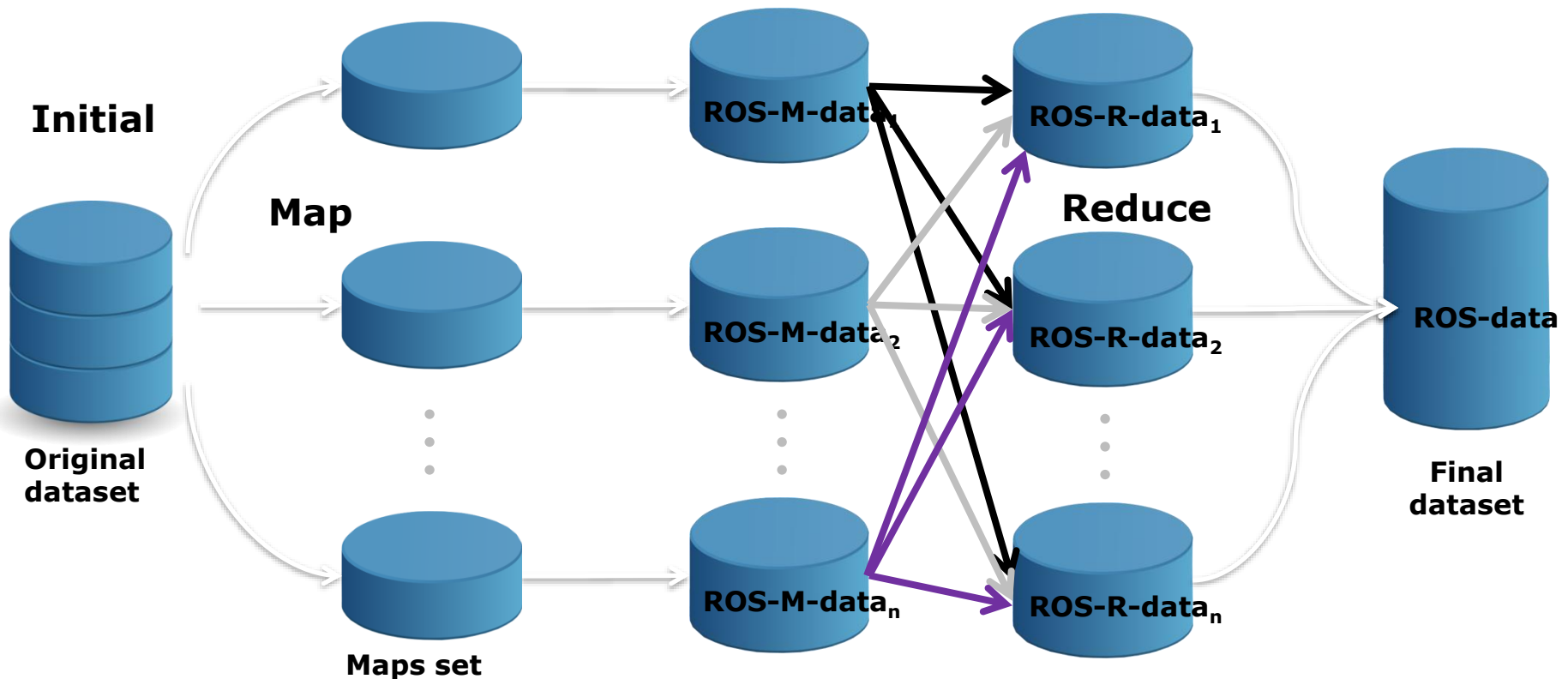


ECBDL'14 Big Data Competition

A MapReduce Approach for Random Oversampling

Low ratio of true contacts ($< 2\%$).

Imbalance rate: > 49 . **Imbalanced problem!**



ECBDL'14 Big Data Competition

We initially focused on

❑ **Oversampling rate: 100%**

RandomForest:

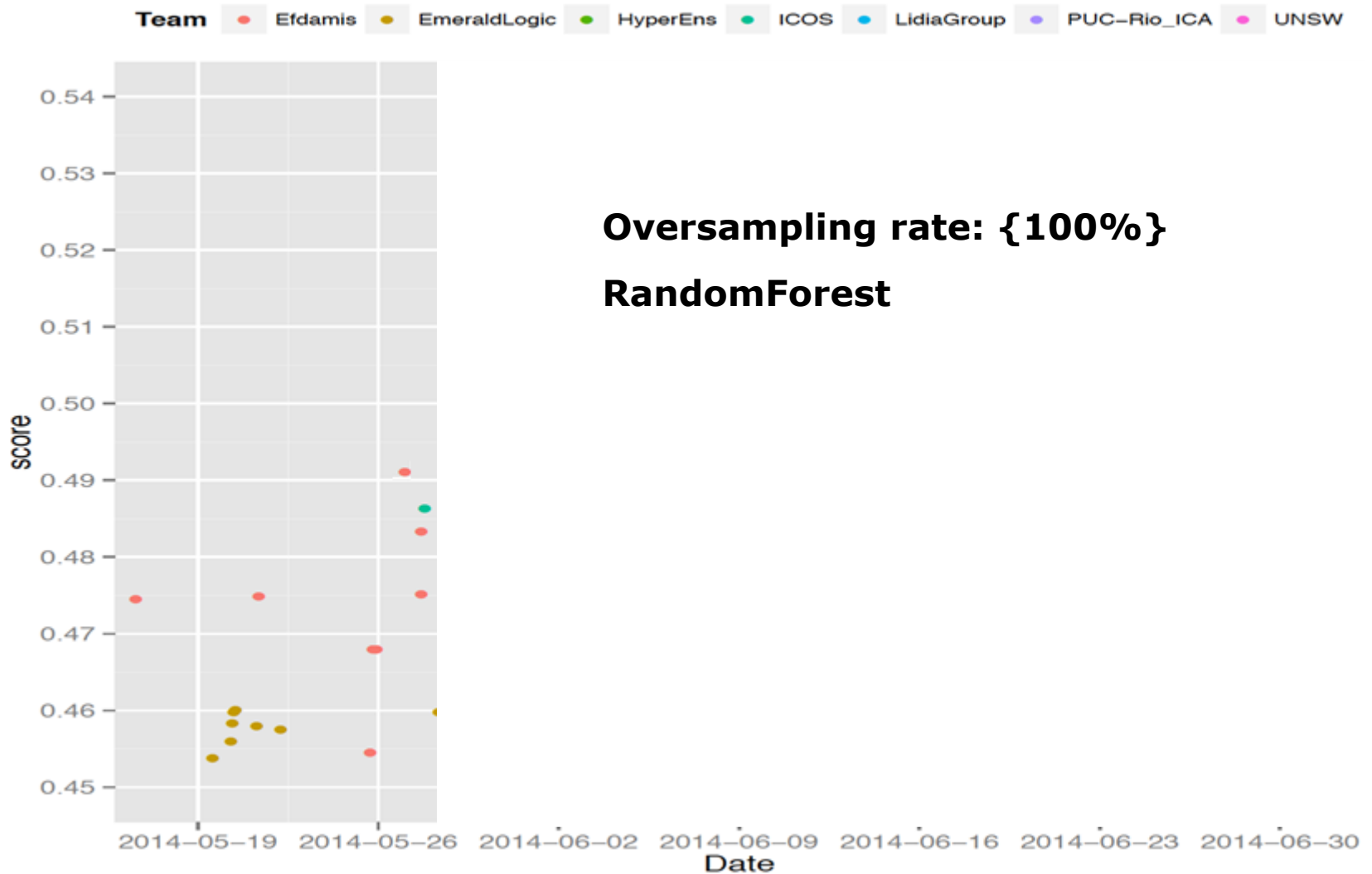
- ❑ Number of used features: $10 (\log n + 1)$; Number of trees: 100
- ❑ Number of maps: {64, 190, 1024, 2048}

| Nº mappers | TPR_tst | TNR_tst | TNR*TPR Test |
|------------|----------|----------|-----------------|
| 64 | 0,601723 | 0,806269 | 0,485151 |

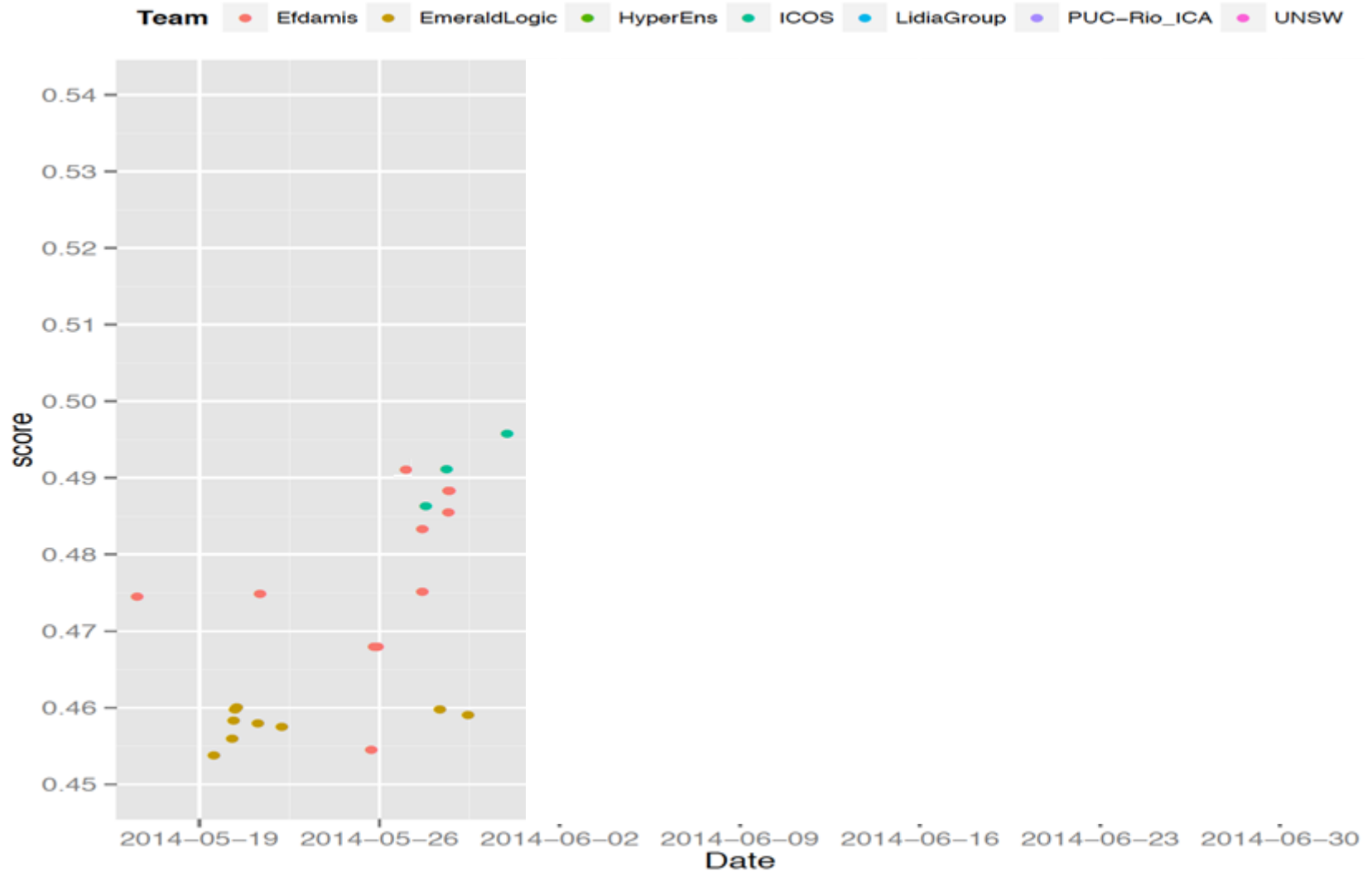
Very low TPR (relevant!)

How to increase the TPR rate?

ECBDL'14 Big Data Competition



ECBDL'14 Big Data Competition



ECBDL'14 Competición Big Data

We initially focused on

❑ **Oversampling rate: 100%**

RandomForest:

- ❑ Number of used features: $10 (\log n + 1)$; Number of trees: 100
- ❑ Number of maps: {64, 190, 1024, 2048}

| Nº mappers | TPR_tst | TNR_tst | TNR*TPR Test |
|------------|----------|----------|-----------------|
| 64 | 0,601723 | 0,806269 | 0,485151 |

Very low TPR (relevant!)

How to increase the TPR rate?

Idea: To increase the ROS porcentaje

ECBDL'14 Big Data Competition

How to increase the TPR rate?

Idea: To increase the ROS porcentaje

- ❑ Oversampling rate: {100, 105, 110, 115, 130}

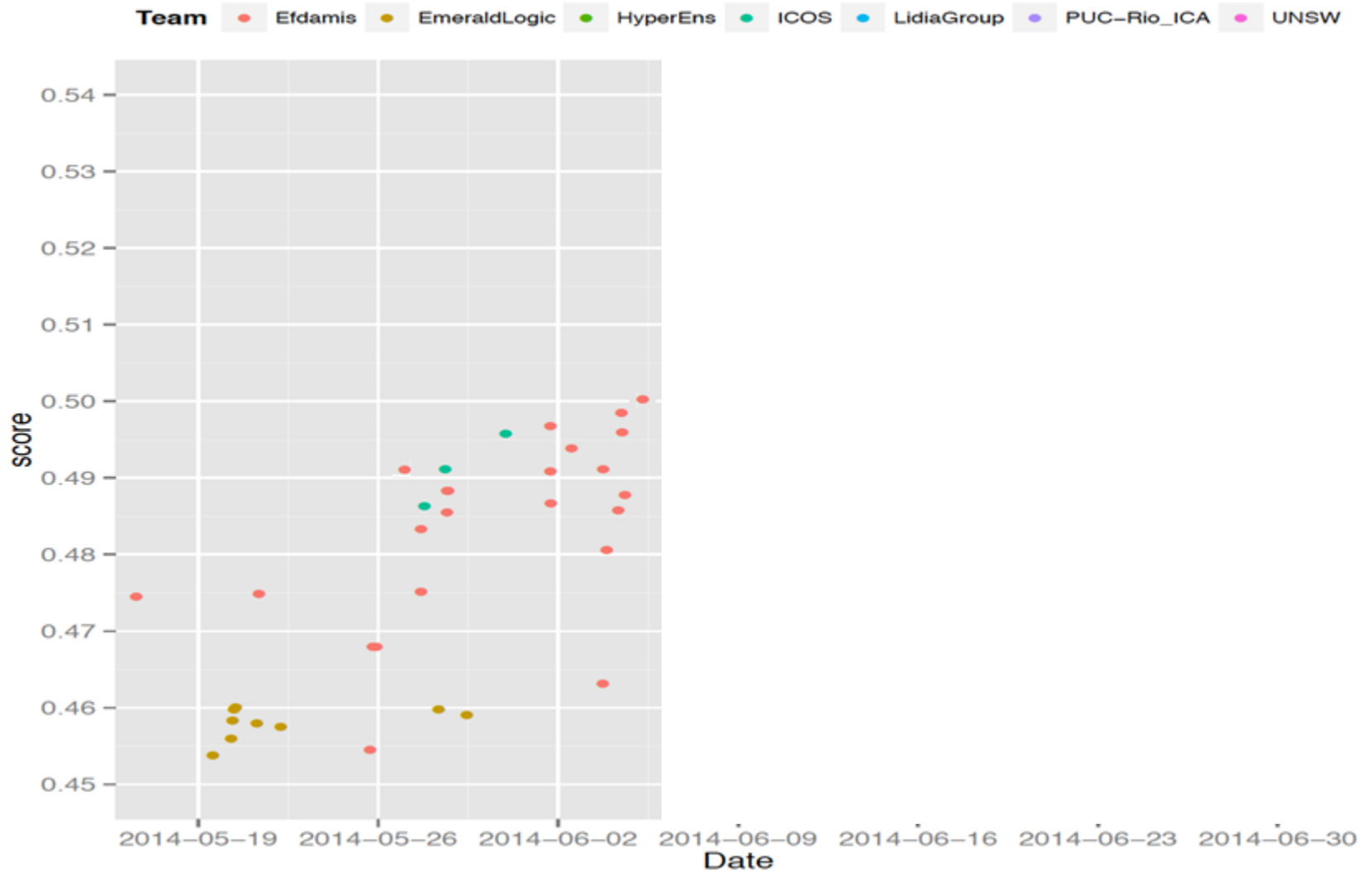
RandomForest:

- ❑ Number of used features:10; Number of trees: 100

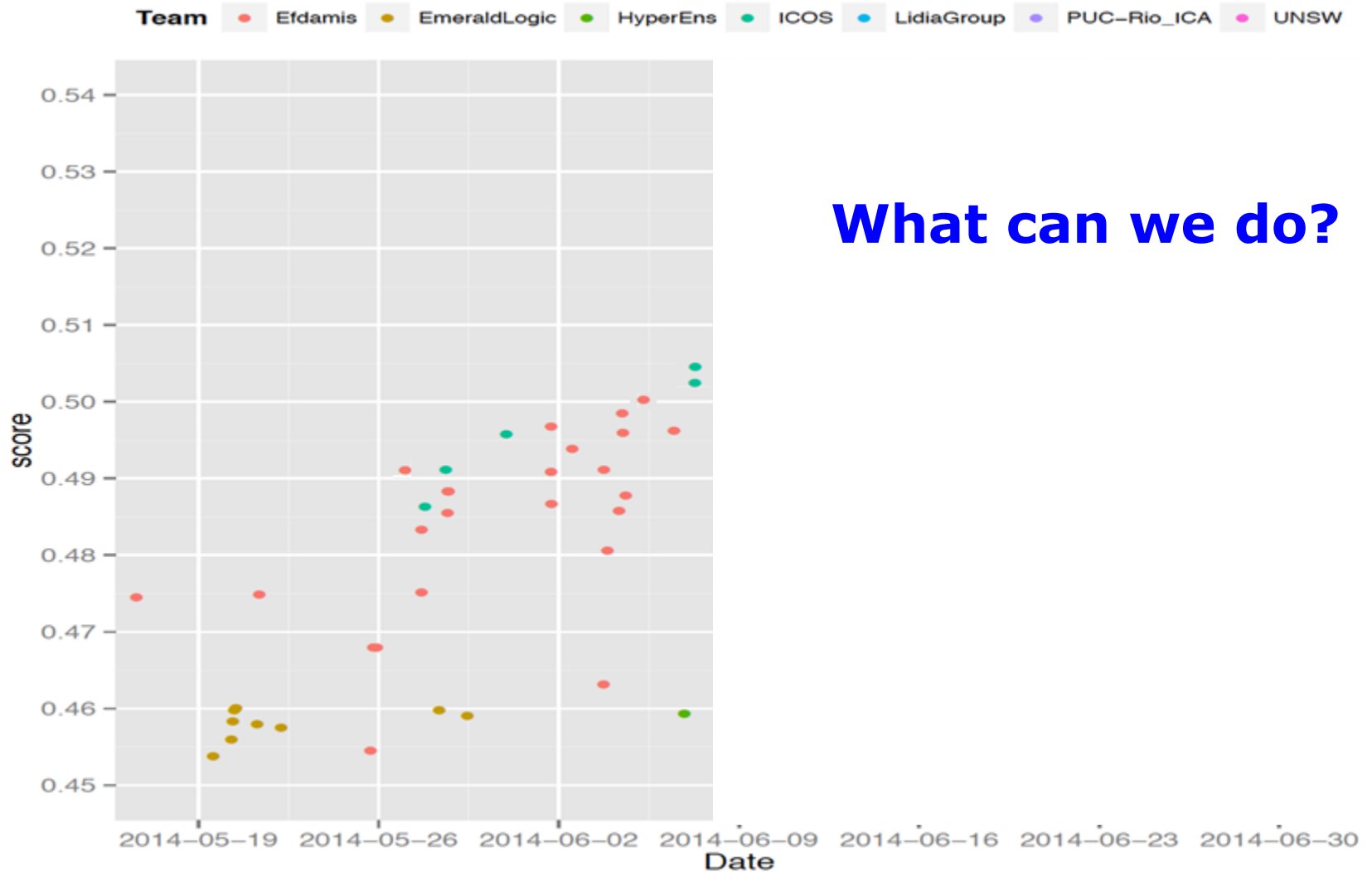
| Algorithms | TPR | TNR | TNR*TPR Test |
|--------------------------|---------------|---------------|-----------------|
| ROS+RF (RS: 100%) | 0.6351 | 0.7733 | 0.491186 |
| ROS+RF (RS: 105%) | 0.6568 | 0.7555 | 0.496286 |
| ROS+RF (RS: 110%) | 0.6759 | 0.7337 | 0.495941 |
| ROS+RF (RS: 115%) | 0.7041 | 0.7103 | 0.500175 |
| ROS+RF (RS: 130%) | 0.7472 | 0.6609 | 0.493913 |

**The higher ROS percentage, the higher TPR
and the lower TNR**

ECBDL'14 Big Data Competition



ECBDL'14 Big Data Competition



What can we do?

ECBDL'14 Big Data Competition

ECBDL'14 Big Data Competition 2014

Our approach:

1. Balance the original training data
 - ❑ Random Oversampling
 - ❑ (As first idea, it was extended)

2. Learning a model.
 - ❑ Random Forest



3. Detect relevant features.
 1. Evolutionary Feature Weighting

Classifying test set.



ECBDL'14 Big Data Competition

How to increase the performance?

Third component: MapReduce Approach for Feature Weighting for getting a major performance over classes

Map Side

- ❑ Each map read one block from dataset.
- ❑ Perform an **Evolutionary Feature Weighting** step.
- ❑ **Output:** a real vector that represents the degree of importance of each feature.
- ❑ Number of maps: 32768 (**less than 1000 original data per map**)

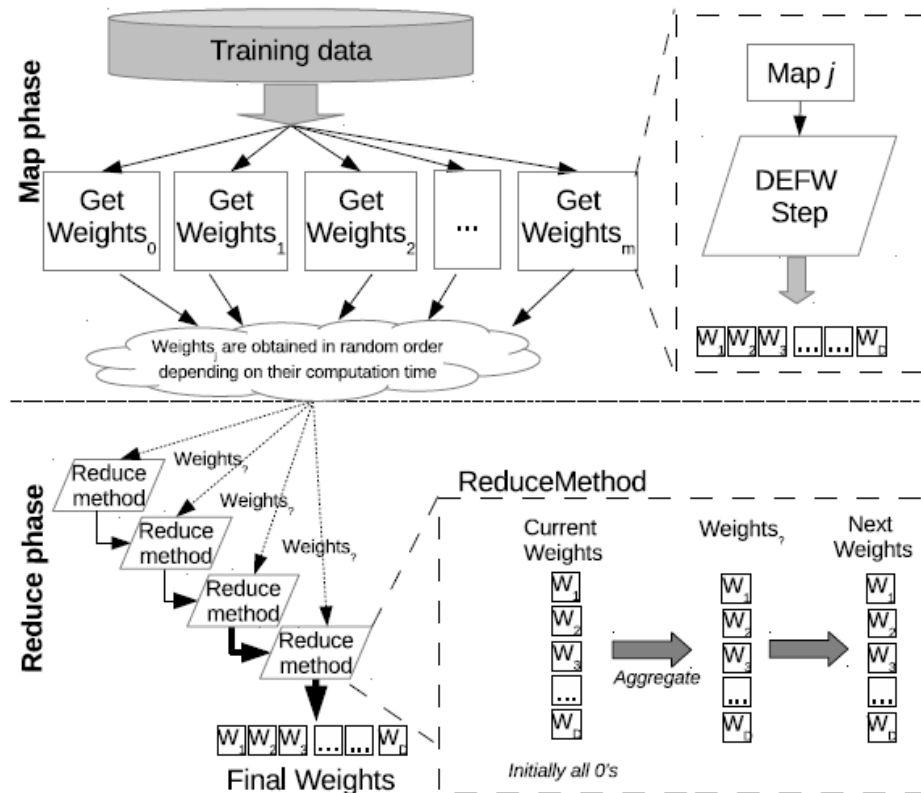
Reduce Side

- ❑ Aggregate the feature's weights
- ❑ A feature is finally selected if it overcomes a given threshold.
- ❑ **Output:** a binary vector that represents the final selection

ECBDL'14 Big Data Competition

How to increase the performance?

Third component: MapReduce Approach for Feature Weighting for getting a major performance over classes



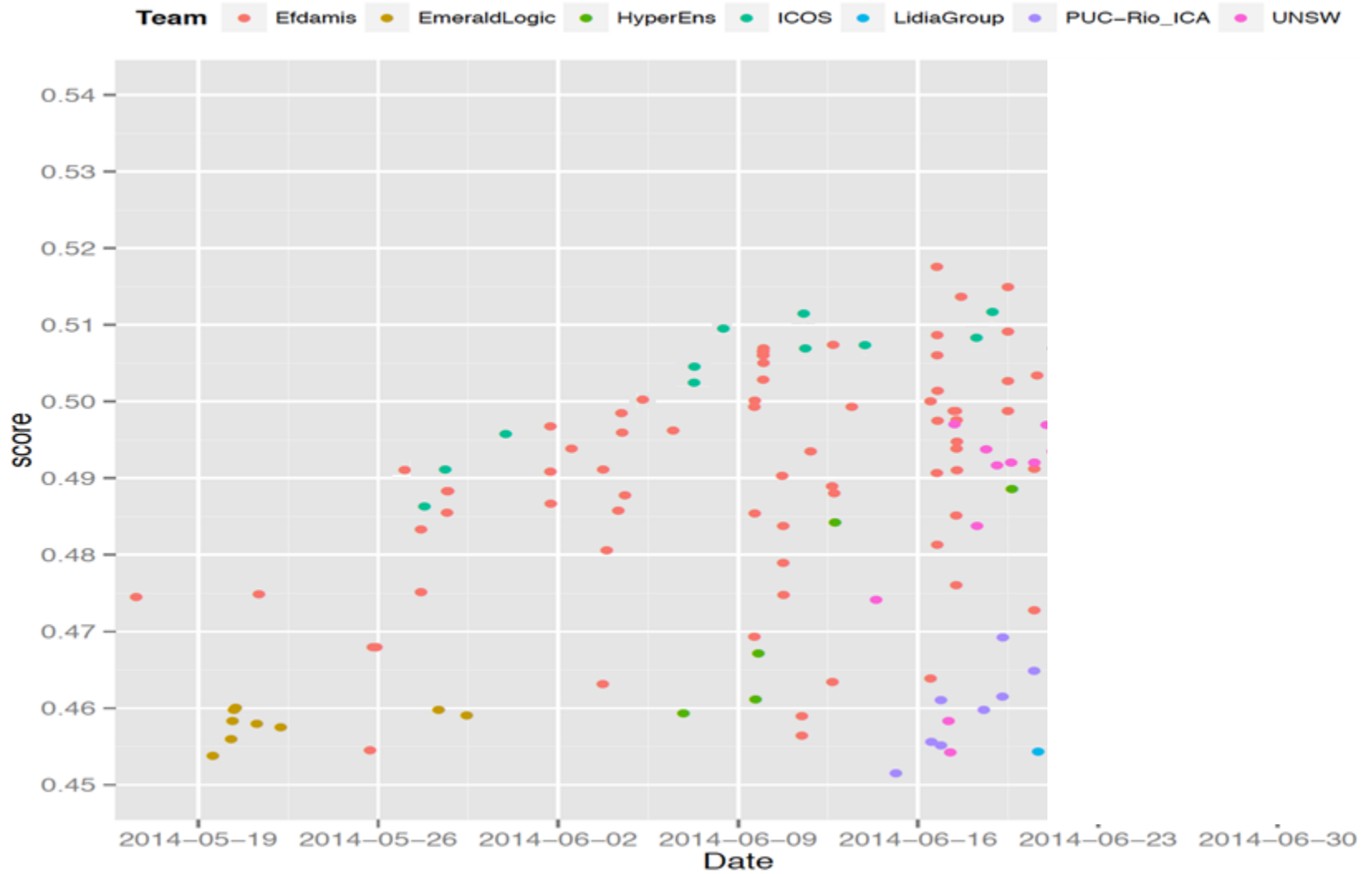
ECBDL'14 Big Data Competition

Evolutionary Feature Weighting.

It allows us to construct several subset of features (changing the threshold).

| Algorithms | 64 mappers | | | |
|--------------------------------------|------------------|----------|----------|--------------|
| | TNR*TPR Training | TPR | TNR | TNR*TPR Test |
| ROS+RF (130% - Feature Weighting 63) | 0.726350 | 0.66949 | 0.775652 | 0.519292 |
| ROS+RF (115% - Feature Weighting 63) | 0.736596 | 0.652692 | 0.790822 | 0.516163 |
| ROS+RF (100% - Feature Weighting 63) | 0.752824 | 0.626190 | 0.811176 | 0.507950 |

ECBDL'14 Big Data Competition



ECBDL'14 Big Data Competition

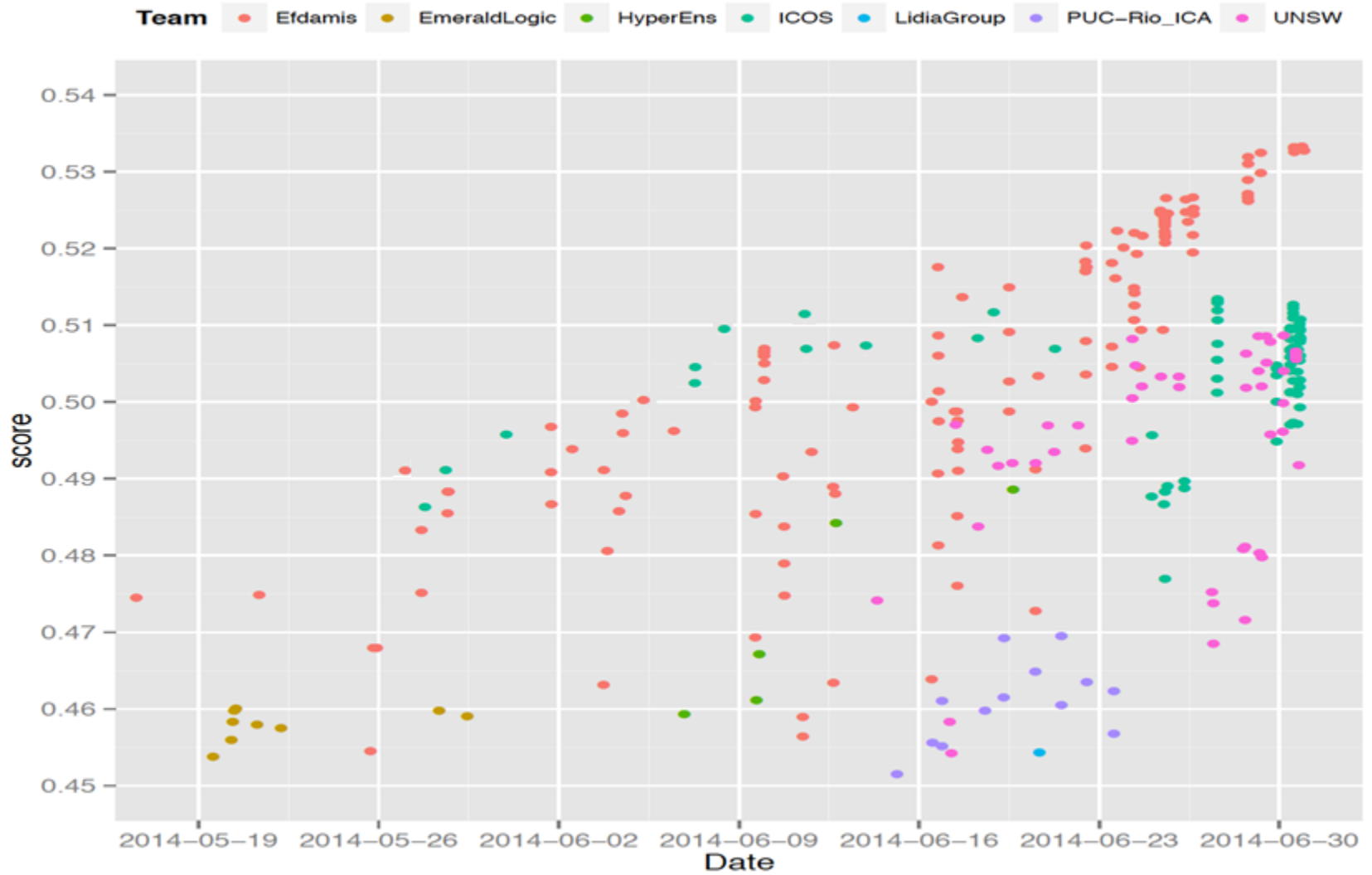
Last decision: We investigated to increase ROS until 180% with 64 mappers

| Algorithms | 64 mappers | | | |
|---------------------------------------|------------------|-----------------|-----------------|-----------------|
| | TNR*TPR Training | TPR | TNR | TNR*TPR Test |
| ROS+ RF (130%+ FW 90+25f+200t) | 0.736987 | 0.671279 | 0.783911 | 0.526223 |
| ROS+ RF (140%+ FW 90+25f+200t) | 0.717048 | 0.695109 | 0.763951 | 0.531029 |
| ROS+ RF (150%+ FW 90+25f+200t) | 0.706934 | 0.705882 | 0.753625 | 0.531971 |
| ROS+ RF (160%+ FW 90+25f+200t) | 0,698769 | 0.718692 | 0.741976 | 0.533252 |
| ROS+ RF (170%+ FW 90+25f+200t) | 0.682910 | 0.730432 | 0.730183 | 0.533349 |
| ROS+ RF (180%+ FW 90+25f+200t) | 0,678986 | 0.737381 | 0.722583 | 0.532819 |

To increase ROS and reduce the mappers number lead us to get a trade-off with good results

ROS 170 – 85 replications of the minority instances

ECBDL'14 Big Data Competition



ECBDL'14 Big Data Competition

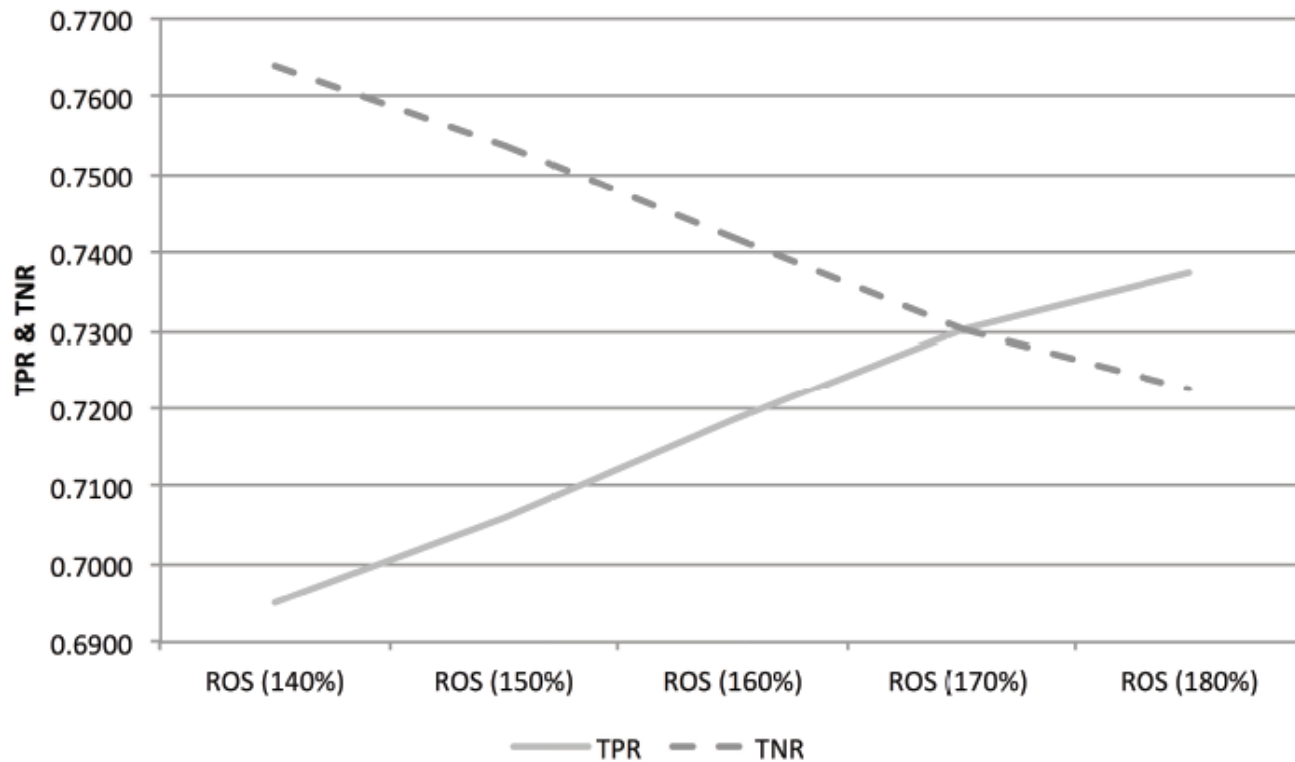


Figure 8: TPR vs. TNR varying the ROS percentage

**ROS 170 – 85 replications of the minority instances
Experiments with 64 maps**

ECBDL'14 Big Data Competition

Evolutionary Computation for Big Data and Big Learning Workshop

Results of the competition: Contact map prediction

| Team Name | TPR | TNR | Acc | TPR · TNR |
|----------------|-----------------|-----------------|-----------------|-----------------|
| Efdamis | 0.730432 | 0.730183 | 0.730188 | 0.533349 |
| ICOS | 0.703210 | 0.730155 | 0.729703 | 0.513452 |
| UNSW | 0.699159 | 0.727631 | 0.727153 | 0.508730 |
| HyperEns | 0.640027 | 0.763378 | 0.761308 | 0.488583 |
| PUC-Rio_ICA | 0.657092 | 0.714599 | 0.713634 | 0.469558 |
| Test2 | 0.632009 | 0.735545 | 0.733808 | 0.464871 |

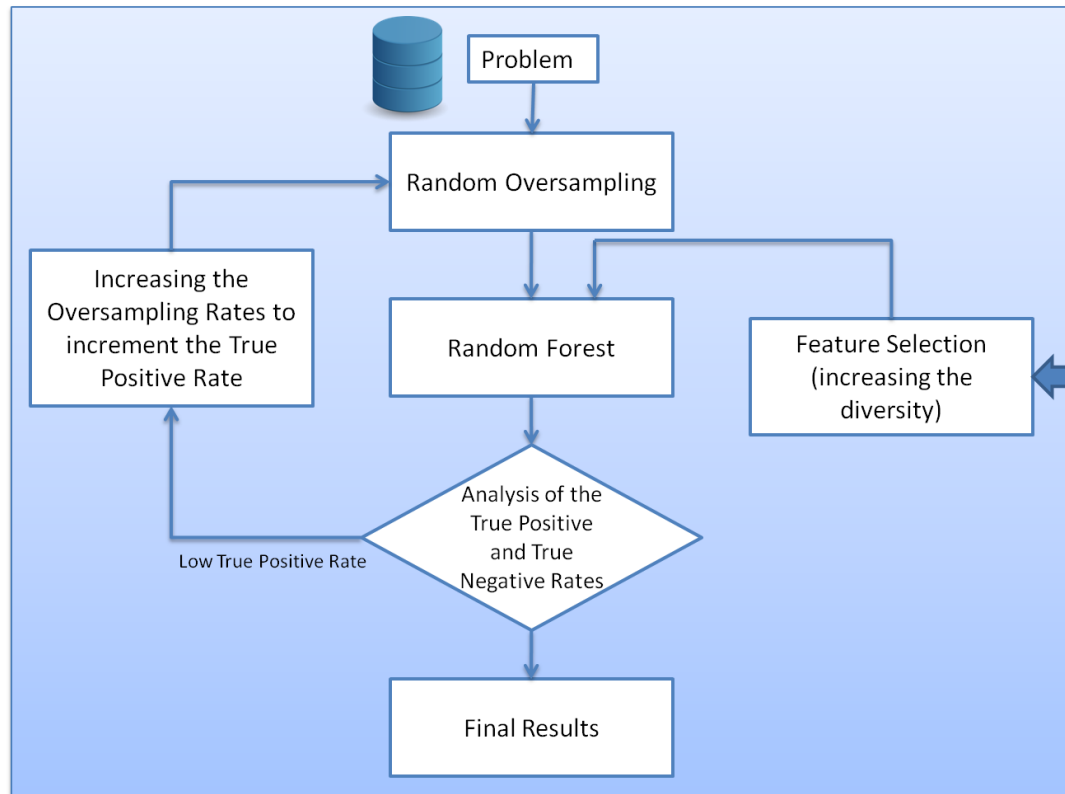
EFDAMIS team ranked first in the ECBDL'14 big data competition

<http://cruncher.ncl.ac.uk/bdcomp/index.pl?action=ranking>

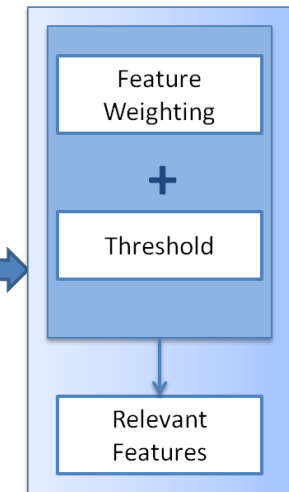
ECBDL'14 Big Data Competition

Our algorithm: **ROSEFW-RF**

**Iterative
MapReduce
Process**



**MapReduce
Process**



I. Triguero, S. del Río, V. López, J. Bacardit, J.M. Benítez, F. Herrera.

ROSEFW-RF: The winner algorithm for the ECBDL'14 Big Data Competition: An extremely imbalanced big data bioinformatics problem.

Knowledge-Based Systems, Volume 87, October 2015, Pages 69–79

<https://github.com/triguero/ROSEFW-RF>

ECBDL'14 Big Data Competition

At the beginning ROS+RF (RS: 100%)

| Nº mappers | TPR_tst | TNR_tst | TNR*TPR Test |
|------------|----------|----------|-----------------|
| 64 | 0,601723 | 0,806269 | 0,485151 |

At the end

| Algorithms | 64 mappers | | | |
|--------------------------------|---------------------|----------|----------|-----------------|
| | TNR*TPR Training | TPR | TNR | TNR*TPR Test |
| ROS+ RF (160%+ FW 90+25f+200t) | 0,698769 | 0.718692 | 0.741976 | 0.533252 |
| ROS+ RF (170%+ FW 90+25f+200t) | 0.682910 | 0.730432 | 0.730183 | 0.533349 |
| ROS+ RF (180%+ FW 90+25f+200t) | 0,678986 | 0.737381 | 0.722583 | 0.532819 |

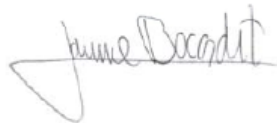


Modelos de calidad están basados en datos de calidad.

ECBDL'14 Big Data Competition

**ECBDL'14: Evolutionary Computation for
Big Data and Big Learning Workshop
July 13th, 2014
GECCO-2014, Vancouver, Canada**

This is to certify that team EFDAMIS, formed
by Isaac Triguero, Sara del Río, Victoria
López, José Manuel Benítez and Francisco
Herrera, ranked **first** in the ECBDL'14 big data
competition



Jaume Bacardit, organizer
ECBDL'14 big data competition



BIG DATA

Índice

- ❑ Big Data. Big Data Science
- ❑ ¿Por qué Big Data? Google crea el Modelo de Programación MapReduce
- ❑ Tecnologías para Big Data: Ecosistema Hadoop (Hadoop, Spark, ...)
- ❑ Big Data Analytics: Librerías para Analítica de Datos en Big Data. Casos de estudio
- ❑ Algunas aplicaciones: Salud, Social Media, Identificación
- ❑ **Comentarios Finales**

Oportunidades en Big Data

Big Data es un área emergente y en expansión. Las posibilidades de desarrollo de algoritmos para nuevos datos, aplicaciones reales ... es un nicho de investigación y desarrollo en los próximos años.



Comentarios Finales



http://elpais.com/elpais/2015/03/26/buena vida/1427382655_646798.html

¿Qué es eso del 'big data'?

- Lo mencionan en conferencias, charlas y facultades. Aclaremos el concepto de moda. O lo que es lo mismo: lo que todas las empresas quieren saber de usted

EVA VAN DEN BERG | 31 MAR 2015 - 12:28 CEST



Tiene continua repercusión en la prensa

Comentarios Finales



- ❑ La paralelización de los algoritmos de aprendizaje automático junto al particionamiento de datos pueden proporcionar algoritmos de calidad con MapReduce.
- ❑ Particionando datos y aplicando el algoritmo a cada parte.
- ❑ Centrando la atención en la fase de combinación (**reduce**). La combinación de modelos es un reto en el diseño de cada algoritmo.
- ❑ Data Mining, Machine learning and data preprocessing: Inmensa colección de algoritmos frente a los pocos algoritmos en big data analytics.

Comentarios Finales



Machine learning: Huge collection of algorithms

Big data: A small subset of algorithms

Para el diseño y/o adaptación de cualquier algoritmo es necesario diseñar de forma adecuada una fase de fusión de información por cuanto siempre será necesario utilizar funciones Map y Reduce cuando la base de datos sea muy grande.

Igualmente los procedimientos iterativos requieren de un diseño adecuado para optimizar la eficiencia.

Todavía se está en una fase muy temprana de diseño de algoritmos de aprendizaje automático para big data.

El preprocesamiento de datos es esencial para mejorar el comportamiento de los algoritmos de aprendizaje. El diseño de estos algoritmos para big data está en una fase muy incipiente.

Comentarios Finales



**Data Mining, Machine learning and data preprocessing:
Inmensa colección de algoritmos**

Big Data Analytics



Big Data: Un pequeño conjunto de algoritmos



**Big Data Preprocessing:
Unos pocos métodos de preprocesamiento.**



**Deep learning:
Redes neuronales para procesamiento
de señales/imágenes en grandes volúmenes.**

Comentarios Finales



Big data and analytics: Un gran reto que ofrece múltiples oportunidades

❑ Pequeño conjunto de algoritmos

Es necesario rediseñar nuevos algoritmos.

❑ Modelo de Computación

- ❑ Precisión y aproximación
- ❑ Requiere “eficiencia” en los algoritmos.

❑ Datos de calidad para modelos de calidad en big data

Modelos/Decisiones de calidad están basados en datos de calidad.

❑ Preprocesamiento en Big Data

❑ Análisis del ruido en datos

Métodos automáticos de limpieza

❑ Procesamiento de valores perdidos

❑ Big Data Reduction

Comentarios Finales



Hacia donde vamos?: 3 etapas de Big Data

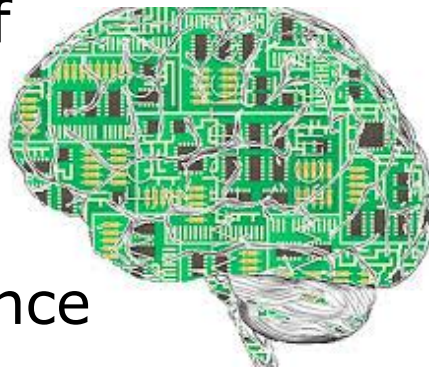
<http://www.kdnuggets.com/2013/12/3-stages-big-data.html>

By Gregory Piatetsky, Dec 8, 2013.



Big Data 3.0: Intelligent Google Now, Watson (IBM) ...

Big Data 3.0 would be a combination of data, with huge knowledge bases and a very large collection of algorithms, perhaps reaching the level of true Artificial Intelligence (Singularity?).



Big Data 1.0:
Transactional



Big Data 2.0:
Networked



Big Data 3.0:
Intelligent

Comentarios Finales



Una demanda creciente de profesionales en "Big Data" y "Ciencia de Datos"

Oportunidades en Big Data (en España)

http://www.revistacloudcomputing.com/2013/10/espana-necesitara-60-000-profesionales-de-big-data-hasta-2015/?goback=.gde_4377072_member_5811011886832984067#!

España necesitará 60.000 profesionales de Big Data hasta 2015

📅 22 octubre, 2013 📍 Eventos 💬 18



España necesitará 60.000 profesionales de Big Data hasta 2015

"España va a necesitar alrededor de sesenta mil profesionales del Big Data de aquí a 2015", así lo ha asegurado Francisco Javier Antón, Subdirector General de Tecnologías del Ministerio de Educación, Cultura y Deportes en una mesa redonda sobre beneficio y aplicación de Big Data en pymes, moderada por Daniel Tapias de [Sigma Technologies](#), celebrada durante el [4º Congreso Nacional de CENTAC](#) de

"Existe una demanda mundial para formar a 4,4 millones de profesionales de la gestión Big Data desde ingenieros, gestores y científicos de datos", comenta Antón. Sin embargo, "las empresas todavía no ven en el Big Data un modelo de negocio", lamenta. "Solo se extrae un 1% de los datos disponibles en la red", añade. "Hace falta formación y concienciación."

Comentarios Finales

**BIG
DATA**



Nubes de datos
Sergio García y Lola Moral
Noviembre 2014

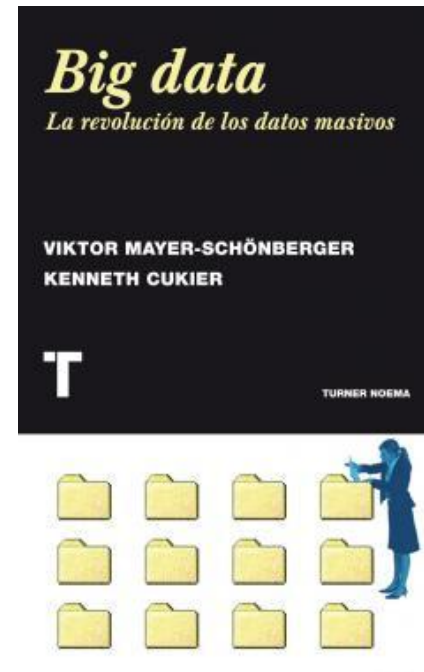
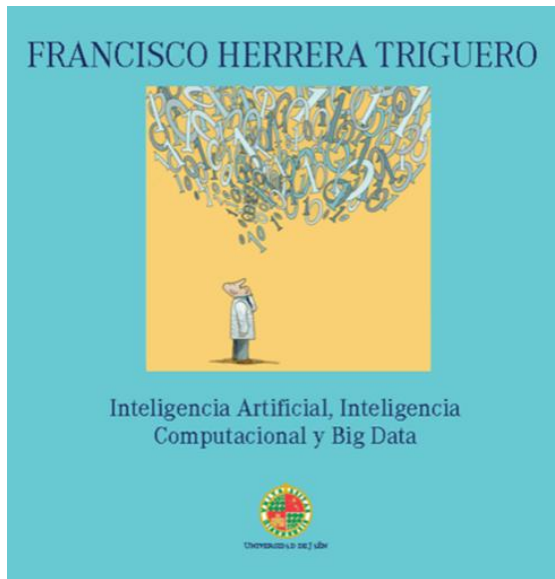
**Las grandes colecciones de datos nos
van a conducir a nuevas innovaciones
(en la industria, ciencia, sociedad)**



Comentarios Finales



2 Lecturas rápidas:



Capítulo 3.

http://issuu.com/secacult_uja/docs/libro_francisco_herrera.indd

A. Fernandez, S. Río, V. López, A. Bawakid, M.J. del Jesus, J.M. Benítez, F. Herrera, **Big Data with Cloud Computing: An Insight on the Computing Environment, MapReduce and Programming Frameworks.** *WIREs Data Mining and Knowledge Discovery* 4:5 (2014) 380-409



¡Gracias!

**BIG
DATA**

Big Data

