

16th IFSA WORLD CONGRESS & 9th EUSFLAT CONFERENCE



Gijón · Spain · June 30th-July 3rd 2015

A tour on big data classification Selected Computational Intelligence approaches



Francisco Herrera

**Research Group on Soft Computing and
Information Intelligent Systems (SCI²S)
Dept. of Computer Science and A.I.
University of Granada, Spain**

Email: herrera@decsai.ugr.es
<http://sci2s.ugr.es>



DECSAI
Universidad de Granada

Big Data

Our world revolves around the data

- **Science**
 - Data bases from astronomy, genomics, environmental data, transportation data, ...
- **Humanities and Social Sciences**
 - Scanned books, historical documents, social interactions data, ...
- **Business & Commerce**
 - Corporate sales, stock market transactions, census, airline traffic, ...
- **Entertainment**
 - Internet images, Hollywood movies, MP3 files, ...
- **Medicine**
 - MRI & CT scans, patient records, ...
- **Industry, Energy, ...**
 - Sensors, ...





Outline

IFSA – EUSFLAT 2015
Celebrating fifty years of Fuzzy Sets

- ❑ Big Data. Big Data Science
- ❑ Why Big Data? MapReduce Paradigm. Hadoop Ecosystem
- ❑ Big Data Classification: Learning algorithms
- ❑ Big Data Classification: Computational Intelligence Approches
- ❑ Big Data Classification: Imbalanced classes
- ❑ Final Comments



Outline

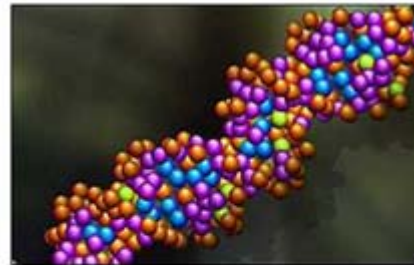
IFSA – EUSFLAT 2015
Celebrating fifty years of Fuzzy Sets

- ❑ Big Data. Big Data Science
- ❑ Why Big Data? MapReduce Paradigm. Hadoop Ecosystem
- ❑ Big Data Classification: Learning algorithms
- ❑ Big Data Classification: Computational Intelligence Approches
- ❑ Big Data Classification: Imbalanced classes
- ❑ Final Comments

What is Big Data? 3 Vs of Big Data



Ej. Genomics



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

Astronomy



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

Transactions



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

What is Big Data? 3 Vs of Big Data



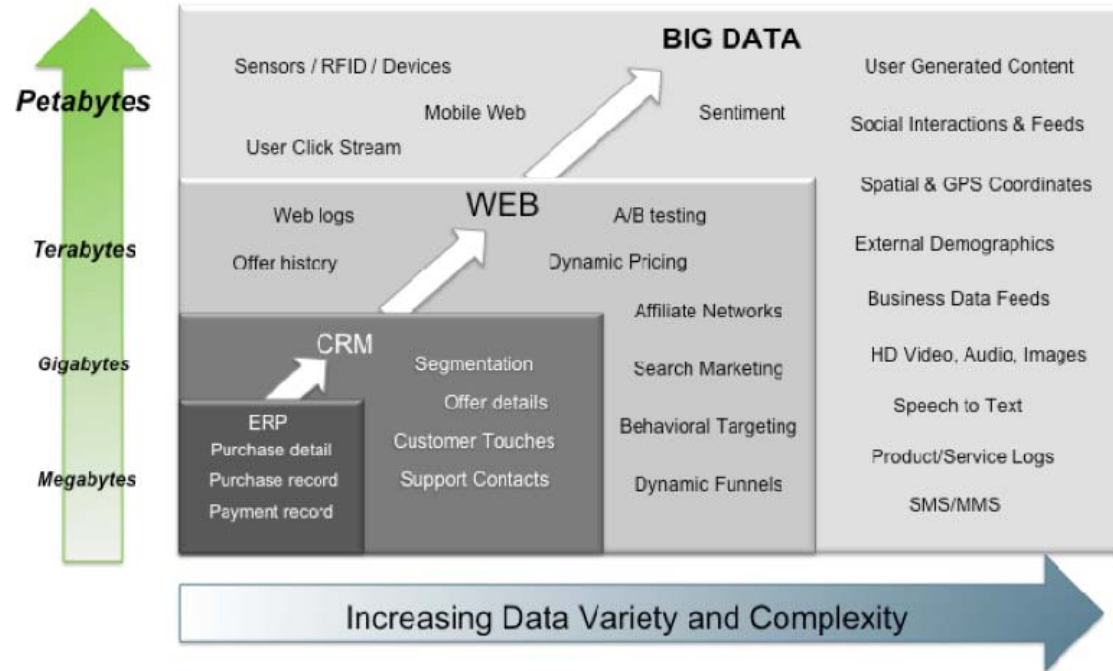
What is Big Data? 3 Vs of Big Data



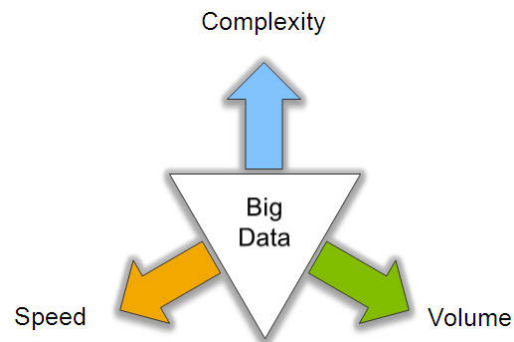
What is Big Data? 3 Vs of Big Data



Big Data = Transactions + Interactions + Observations

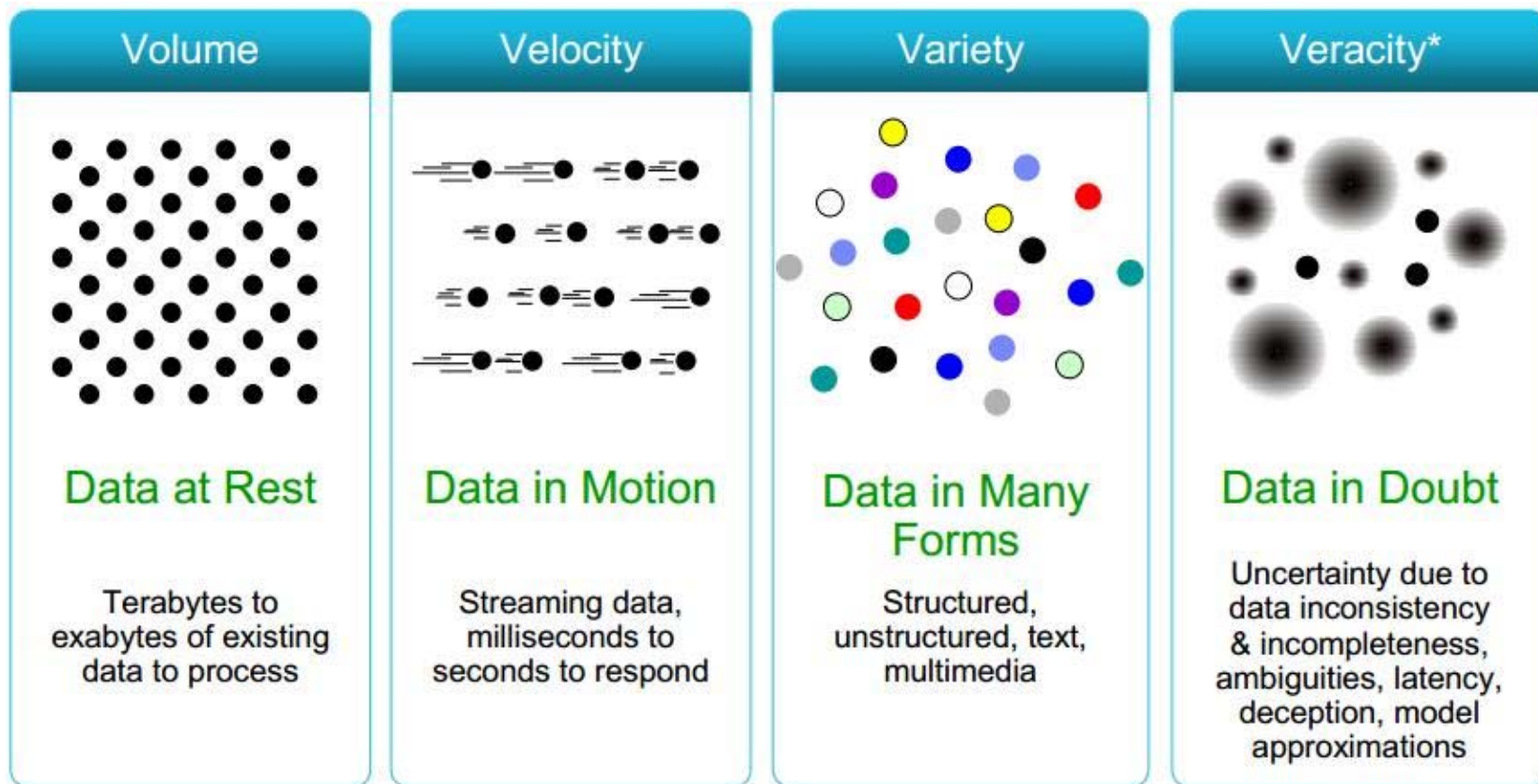


Source: Contents of above graphic created in partnership with Teradata, Inc.



What is Big Data? 3 Vs of Big Data

Some Make it 4V's



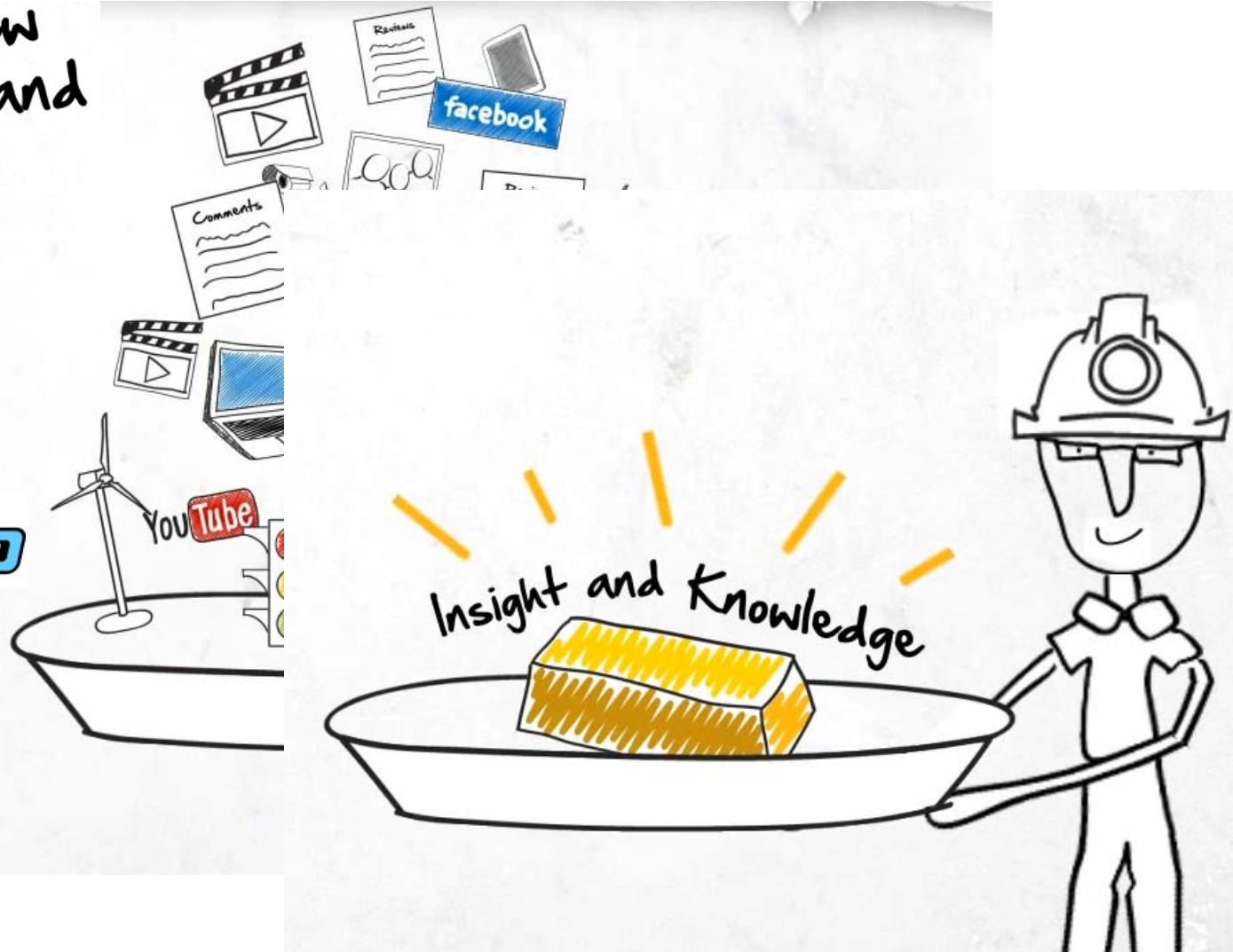
What is Big Data?

5 V's --> Value

Innovative new approaches and technologies



MapReduce



What is Big Data?

No single standard definition



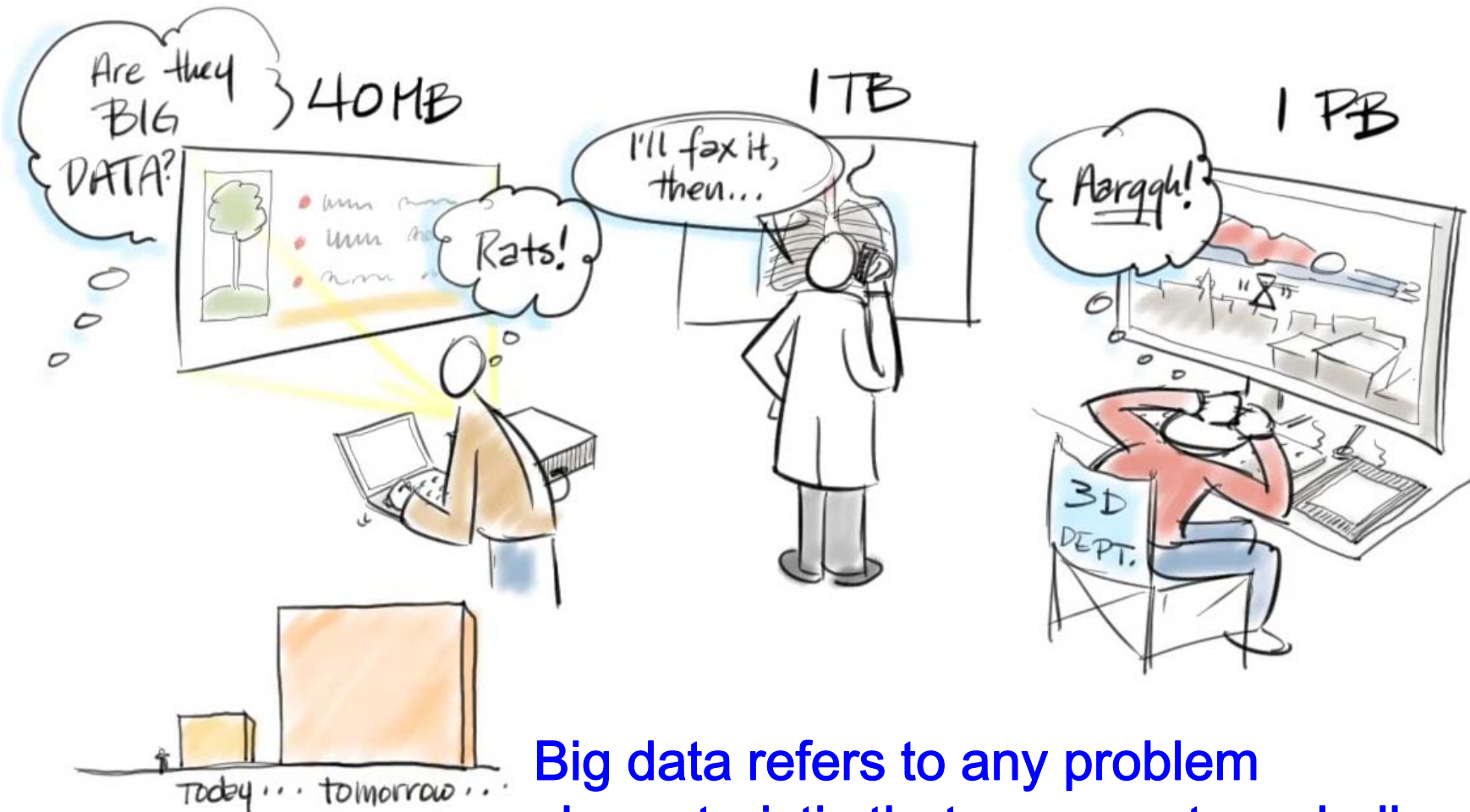
Big data is a collection of data sets so large and complex that it becomes difficult to process using on-hand database management tools or traditional data processing applications.



"Big Data" is data whose scale, diversity, and complexity **require new architectures, techniques, algorithms, and analytics** to manage it and extract value and hidden knowledge from it...



What is Big Data? (in short)



Big data refers to any problem characteristic that represents a challenge to process it with traditional applications

What is Big Data?

Who's Generating Big Data?



Social media and networks
(all of us are generating data)



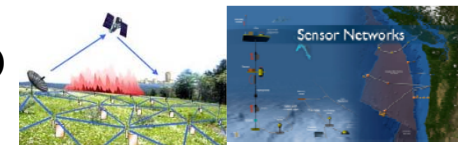
Scientific instruments
(collecting all sorts of data)



Mobile devices
(tracking all objects
all the time)



Transactions



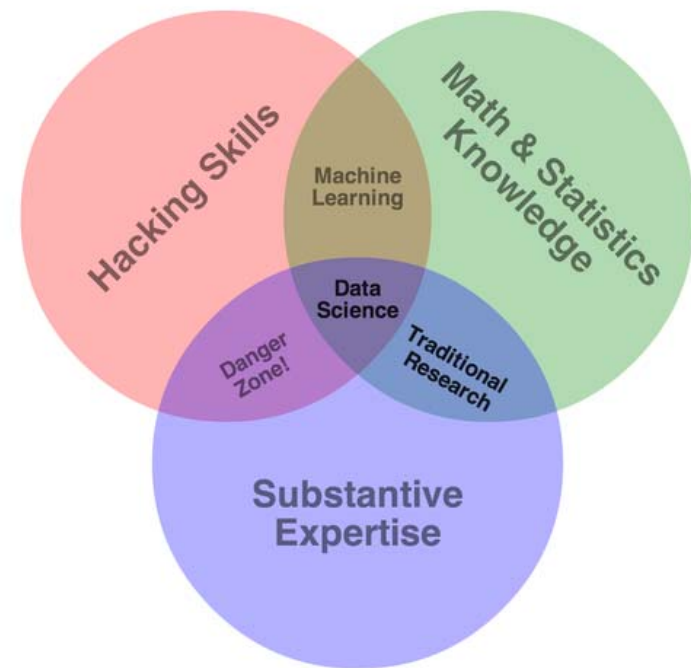
**Sensor technology and
networks**
(measuring all kinds of data)

The progress and innovation is no longer hindered by the ability to collect data but, by the ability to manage, analyze, summarize, visualize, and discover knowledge from the collected data in a timely manner and in a scalable fashion

Big Data Science

Data Science combines the traditional scientific method with the ability to munch, 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
- ...

What is Big Data? Example

ECBDL'14 Big Data Competition 2014 (GEGGO 2014, Vancouver)

Objective: Contact map prediction

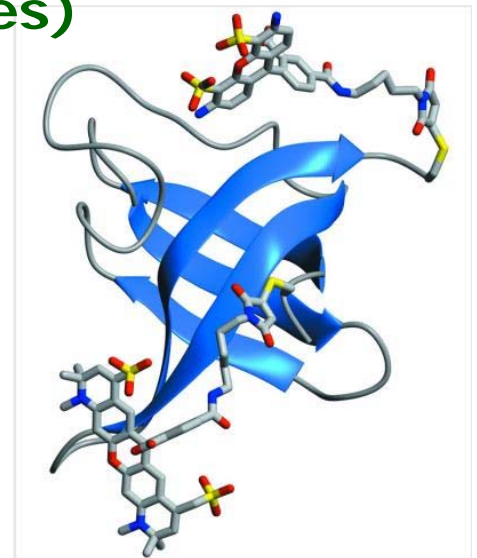
Details:

- ❑ 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





Outline

IFSA – EUSFLAT 2015
Celebrating fifty years of Fuzzy Sets

- Big Data. Big Data Science
- Why Big Data? MapReduce Paradigm. Hadoop Ecosystem
- Big Data Classification: Learning algorithms
- Big Data Classification: Computational Intelligence Approches
- Big Data Classification: Imbalanced classes
- Final Comments

What is Big Data? Why Big Data?

- Scalability to large data volumes:
 - Scan 100 TB on 1 node @ 50 MB/sec = 23 days
 - Scan on 1000-node cluster = 33 minutes
- ➔ Divide-And-Conquer (i.e., data partitioning)



A single machine can not manage large volumes of data efficiently

Why Big Data? MapReduce



- Scalability to large data volumes:
 - Scan 100 TB on 1 node @ 50 MB/sec = 23 days
 - Scan on 1000-node cluster = 33 minutes
- Divide-And-Conquer (i.e., data partitioning)

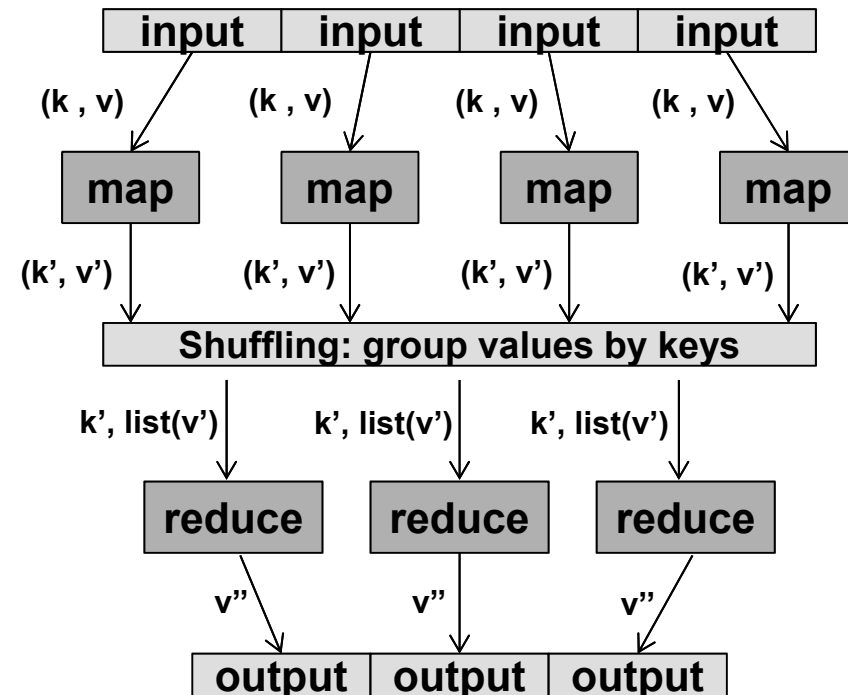
MapReduce

- **Overview:**
 - Data-parallel programming model
 - An associated parallel and distributed implementation for commodity clusters
- **Pioneered by Google**
 - Processes 20 PB of data per day
- **Popularized by open-source Hadoop project**
 - Used by Yahoo!, Facebook, Amazon, and the list is growing ...

MapReduce



- MapReduce is a popular approach to deal with Big Data
- Based on a *key-value pair* data structure
- Two key operations:
 1. **Map function:** Process independent data blocks and outputs summary information
 2. **Reduce function:** Further process previous independent results



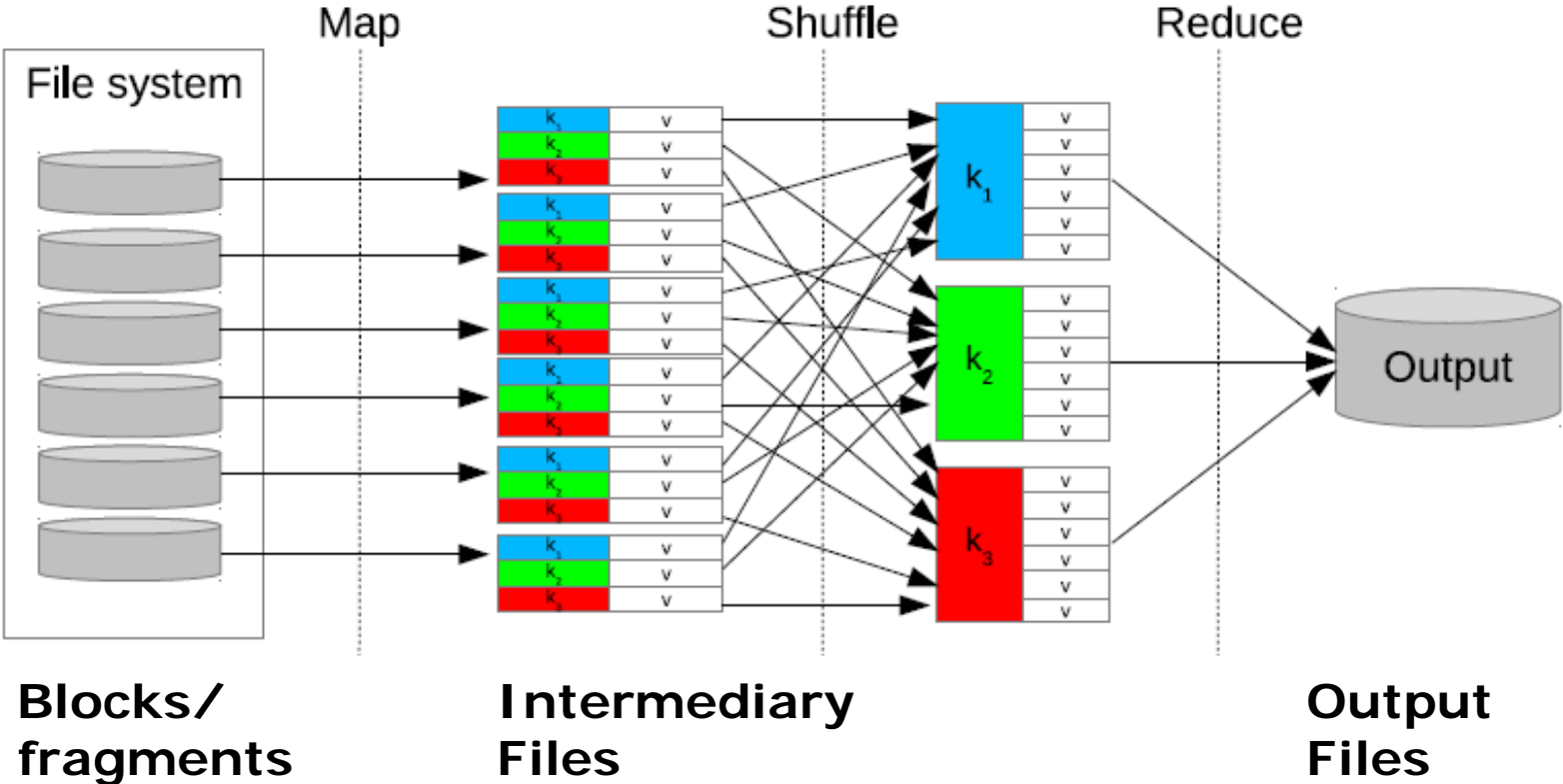
$\text{map } (k, v) \rightarrow \text{list } (k', v')$
 $\text{reduce } (k', \text{list}(v')) \rightarrow v''$

J. Dean, S. Ghemawat, MapReduce: Simplified data processing on large clusters, Communications of the ACM 51 (1) (2008) 107-113.

MapReduce

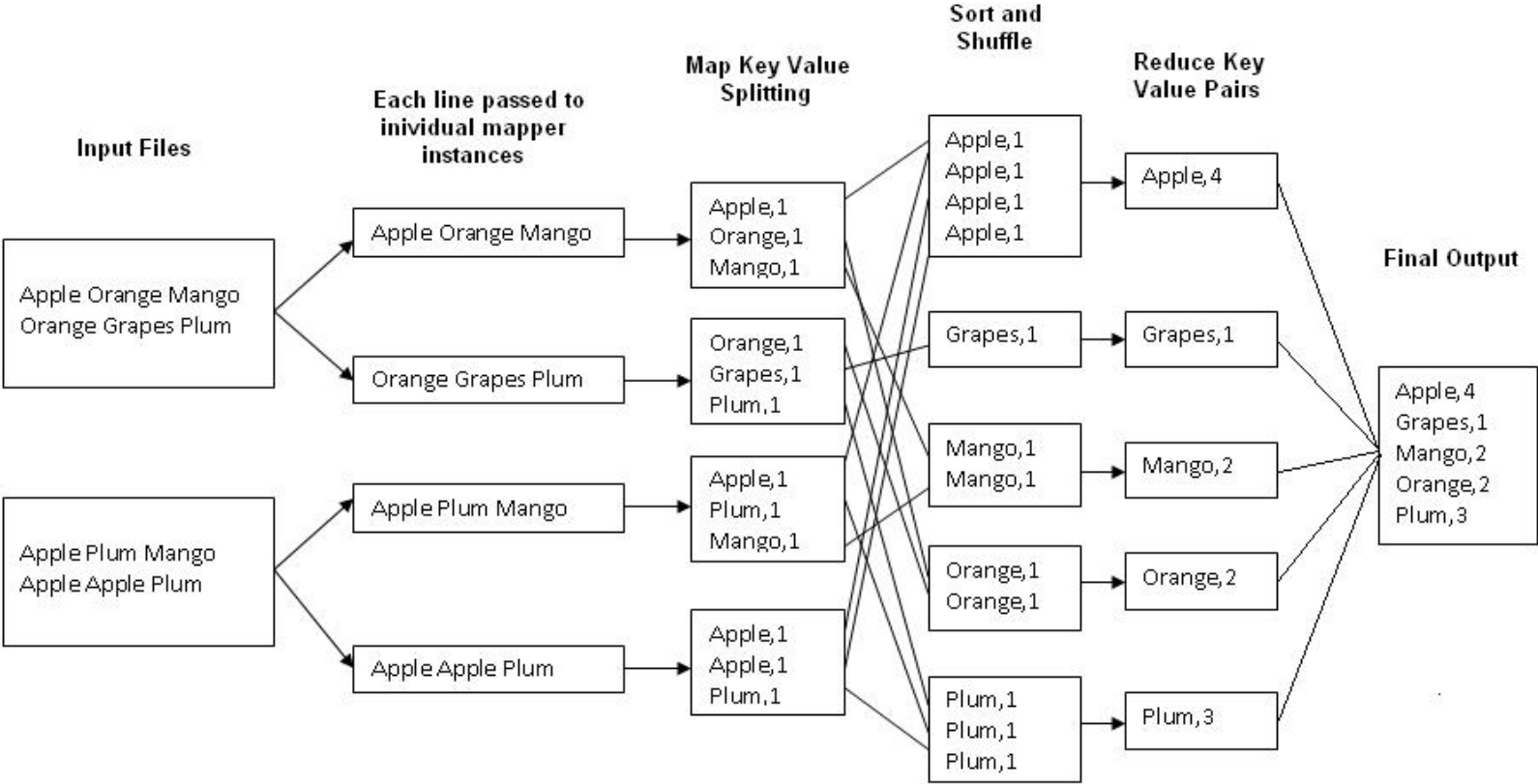


MapReduce workflow



The key of a MapReduce data partitioning approach is usually on the **reduce phase**

MapReduce



MapReduce



Experience

- **Runs on large commodity clusters:**
 - 10s to 10,000s of machines
- **Processes many terabytes of data**
- **Easy to use since run-time complexity hidden from the users**
- **Cost-efficiency:**
 - Commodity nodes (cheap, but unreliable)
 - Commodity network
 - Automatic fault-tolerance (fewer administrators)
 - Easy to use (fewer programmers)

MapReduce



- **Advantage:** MapReduce's data-parallel programming model hides complexity of distribution and fault tolerance
- **Key philosophy:**
 - *Make it scale*, so you can throw hardware at problems
 - *Make it cheap*, saving hardware, programmer and administration costs (but requiring fault tolerance)
- MapReduce **is not suitable for all problems, but when it works, it may save you a lot of time**

MapReduce. Hadoop



Hadoop is an open
source
implementation of
MapReduce
computational
paradigm



Created by **Doug Cutting**
(chairman of board of
directors of the Apache
Software Foundation, 2010)



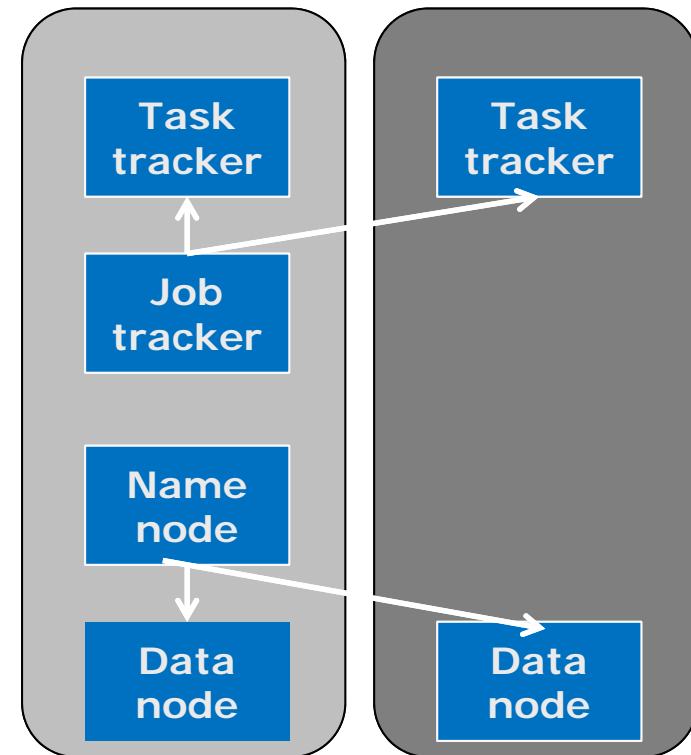
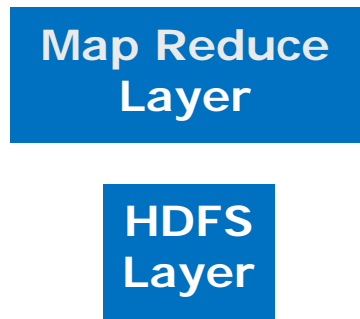
<http://hadoop.apache.org/>

Hadoop

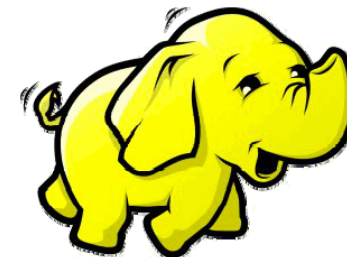


Apache Hadoop is an open-source software framework that supports data-intensive distributed applications, licensed under the Apache v2 license.

Hadoop implements the computational paradigm named MapReduce.



Created by **Doug Cutting** (chairman of board of directors of the Apache Software Foundation, 2010)



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

Hadoop



How do I access to a Hadoop platform?

Cloud Platform
with Hadoop
installation

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



Windows Azure™



Windows Azure

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

Cluster Instalation

Example ATLAS, SCI²S Research Group



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)

Hadoop birth

July 2008 - Hadoop Wins Terabyte Sort Benchmark

One of Yahoo's Hadoop clusters sorted 1 terabyte of data in 209 seconds, which beat the previous record of 297 seconds in the annual general purpose (Daytona) **terabyte short benchmark**. This is the first time that either a **Java** or an open source program has won.

2008, 3.48 minutes

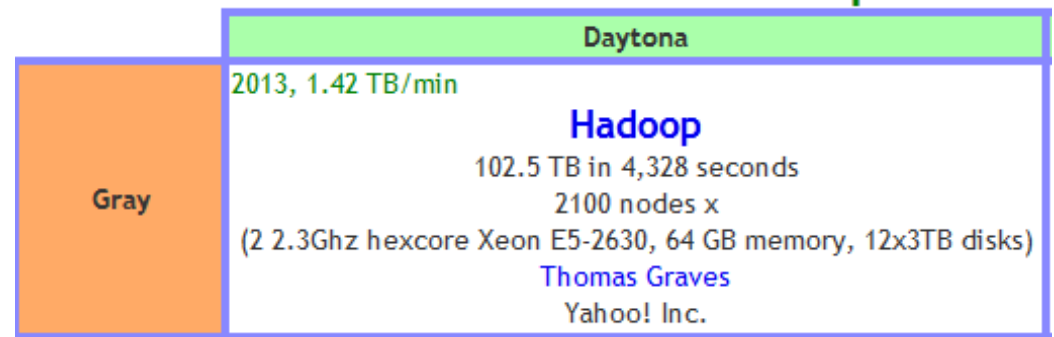
Hadoop

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



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

Hadoop Ecosystem



The project

The project includes these modules:

- **Hadoop Common:** The common utilities that support the other Hadoop modules.
- **Hadoop Distributed File System (HDFS™):** A distributed file system that provides high-throughput access to application data.
- **Hadoop YARN:** A framework for job scheduling and cluster resource management.
- **Hadoop MapReduce:** A YARN-based system for parallel processing of large data sets.

Other Hadoop-related projects at Apache include:

- [Avro™](#): A data serialization system.
- [Cassandra™](#): A scalable multi-master database with no single points of failure.
- [Chukwa™](#): A data collection system for managing large distributed systems.
- [HBase™](#): A scalable, distributed database that supports structured data storage for large tables.
- [Hive™](#): A data warehouse infrastructure that provides data summarization and ad hoc querying.
- [Mahout™](#): A Scalable machine learning and data mining library.
- [Pig™](#): A high-level data-flow language and execution framework for parallel computation.
- [ZooKeeper™](#): A high-performance coordination service for distributed applications.

Recently: [Apache Spark](#)



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

MapReduce: Limitations

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

MAPREDUCE
IS GOOD
ENOUGH?



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

The following malfunctions types of algorithms are examples where MapReduce:

Iterative Graph Algorithms: PageRank
Gradient Descent
Expectation Maximization

Pregel (Google)

Pregel: A System for Large-Scale Graph Processing

Hadoop

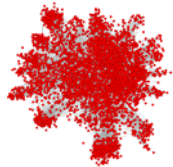
On the limitations of Hadoop. New platforms



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



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



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



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



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



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



Spark (UC Berkeley)
(100 times more efficient than Hadoop, including iterative algorithms, according to creators)
<http://spark.incubator.apache.org/research.html>

GPU based platforms

Mars
GreX



MapReduce

More than 10000 applications in Google

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](#)

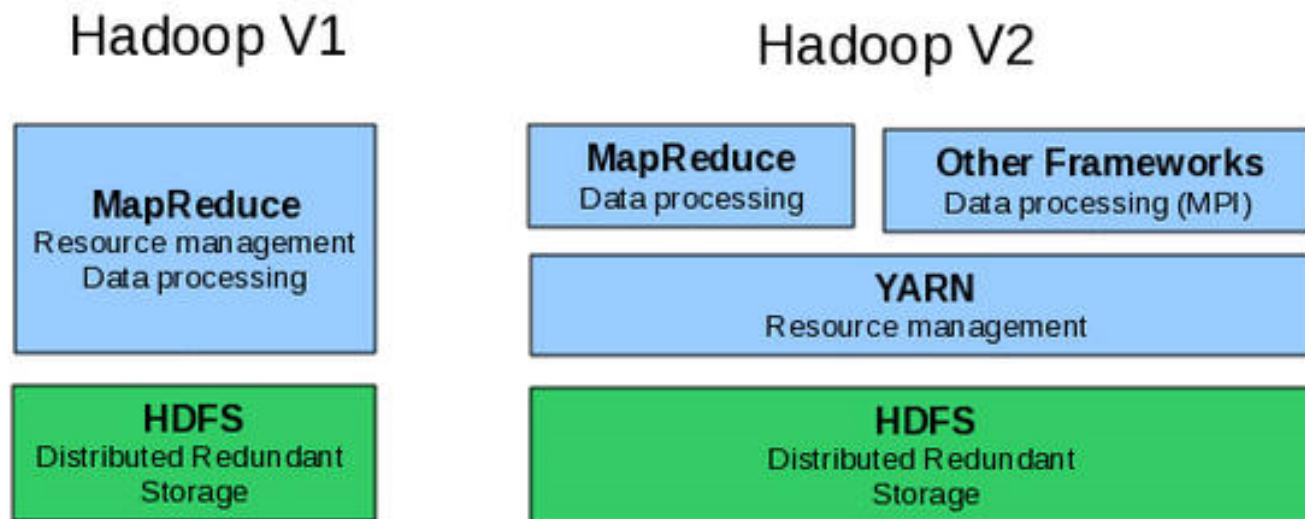


Enrique Alfonseca
Google Research Zurich

Hadoop Evolution



MapReduce Limitations. Graph algorithms (Page Rank, Google), iterative algorithms.



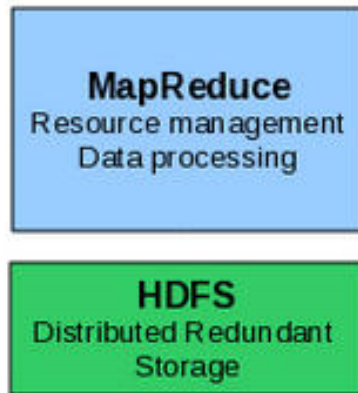
Hadoop Ecosystem

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

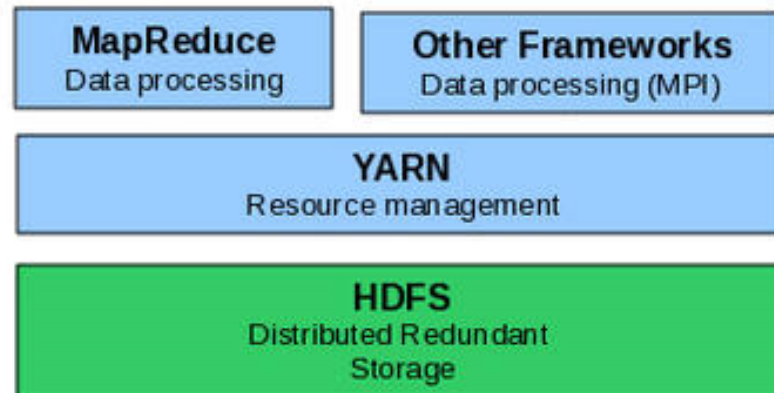
Apache Spark



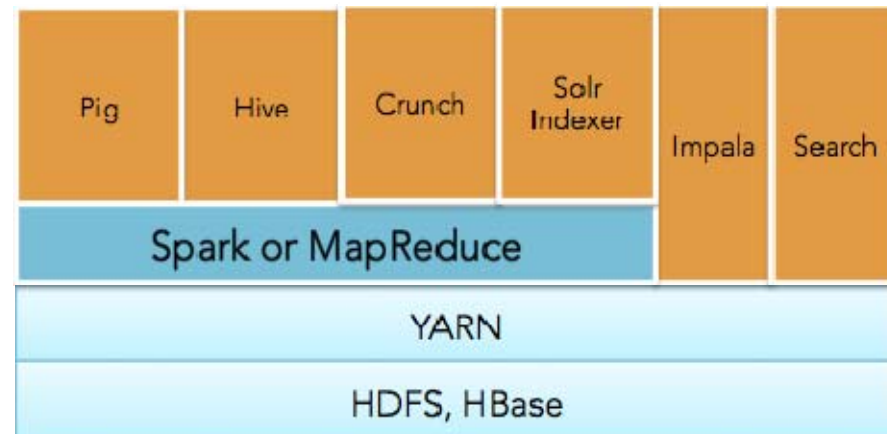
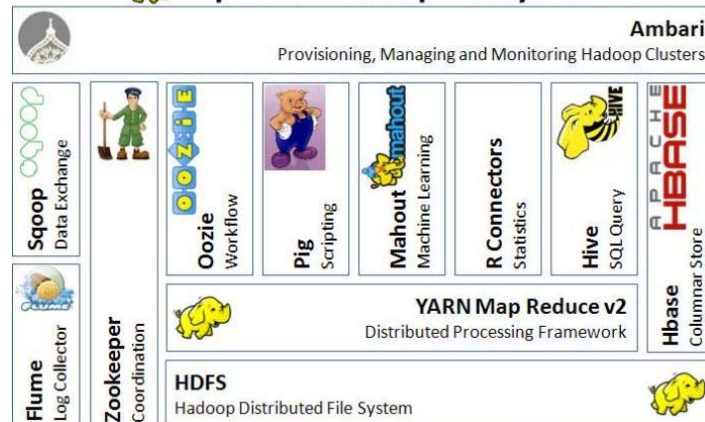
Hadoop V1



Hadoop V2



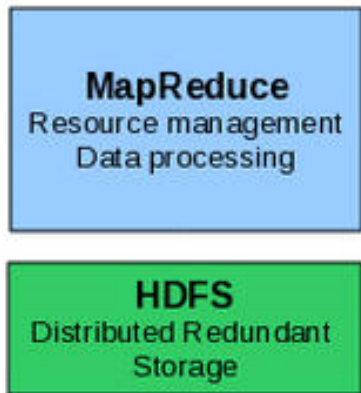
Apache Hadoop Ecosystem



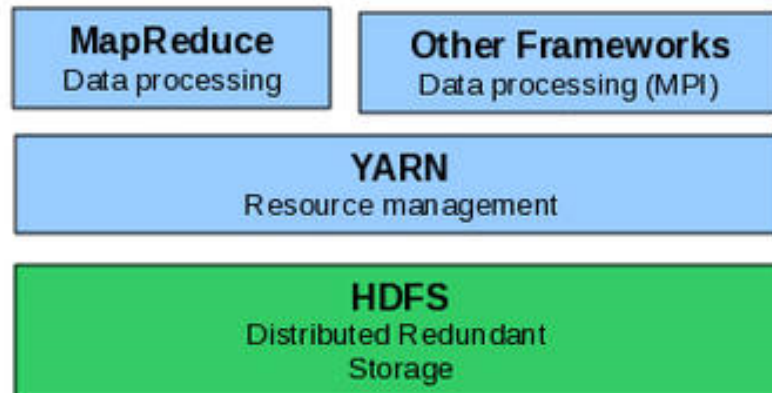
Apache Spark: InMemory



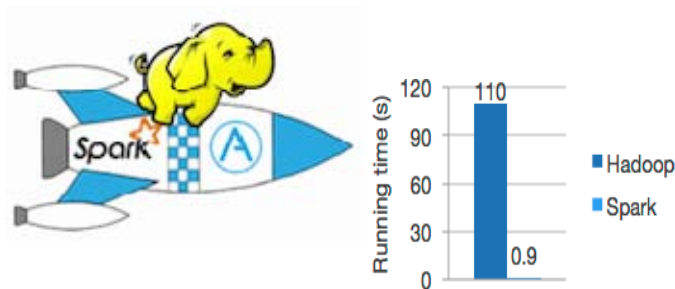
Hadoop V1



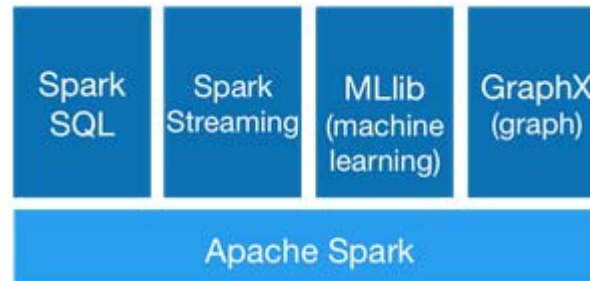
Hadoop V2



InMemory HDFS Hadoop + SPARK



Ecosystem Apache Spark



Future version of Mahout for Spark



Spark birth



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.

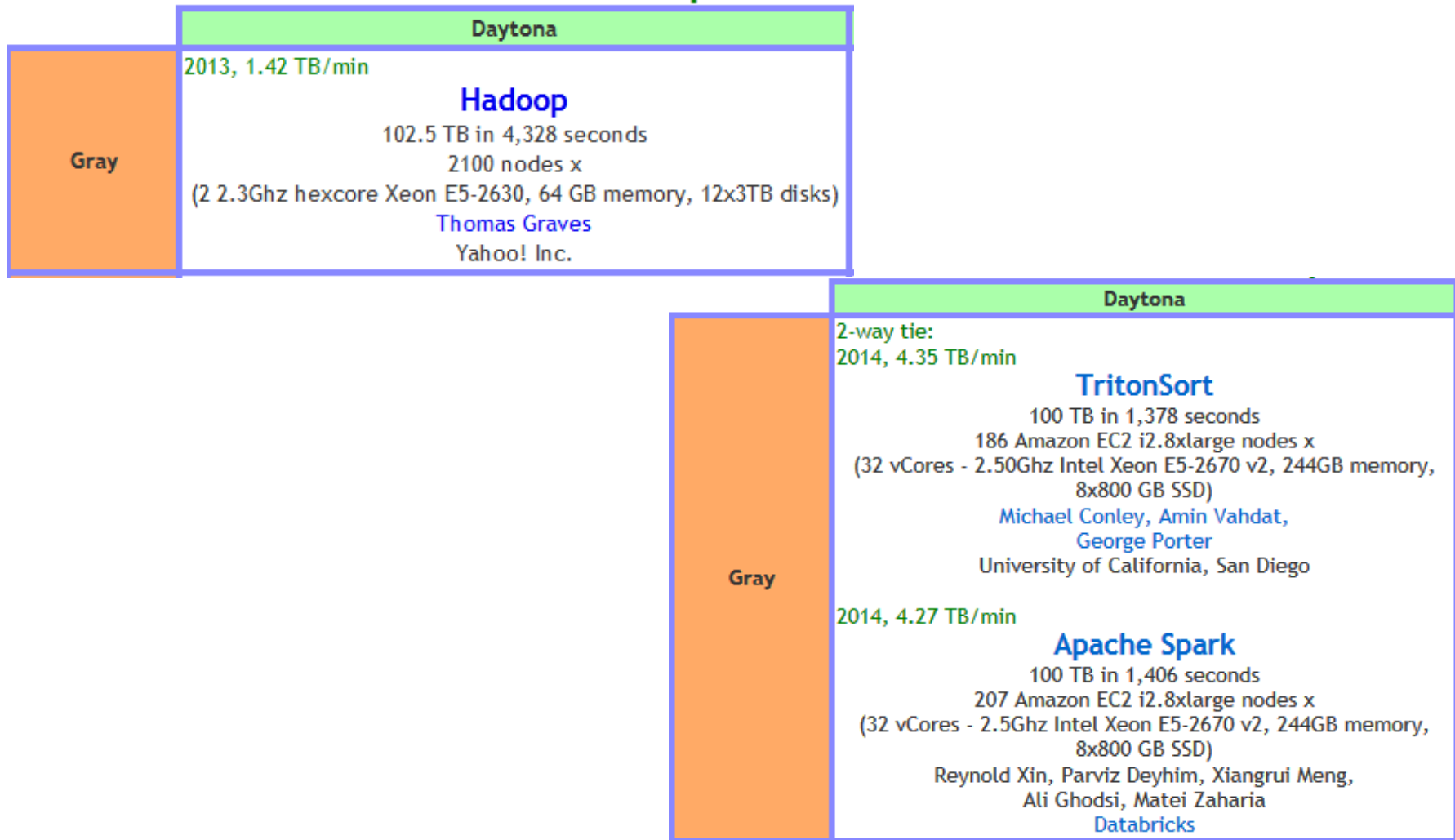
Gray

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.**

	Hadoop World Record	Spark 100 TB *
Data Size	102.5 TB	100 TB
Elapsed Time	72 mins	23 mins
Rate	1.42 TB/min	4.27 TB/min

Spark birth



<http://sortbenchmark.org/>



Outline

IFSA – EUSFLAT 2015
Celebrating fifty years of Fuzzy Sets

- ❑ Big Data. Big Data Science
- ❑ Why Big Data? MapReduce Paradigm. Hadoop Ecosystem
- ❑ **Big Data Classification: Learning algorithms**
- ❑ Big Data Classification: Computational Intelligence Approches
- ❑ Big Data Classification: Imbalanced classes
- ❑ Final Comments

Classification

Generation	1st Generation	2nd Generation	3rd Generation
Examples	SAS, R, Weka, SPSS, KEEL	Mahout, Pentaho, Cascading	Spark, Haloop, 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

Classification

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 - Classification and Regression

MLlib supports various methods for [binary classification](#), [multiclass classification](#), and [regression analysis](#). The table below outlines the supported algorithms for each type of problem.

Problem Type	Supported Methods
Binary Classification	linear SVMs, logistic regression, decision trees, random forests, gradient-boosted trees, naive Bayes
Multiclass Classification	decision trees, random forests, naive Bayes
Regression	linear least squares, Lasso, ridge regression, decision trees, random forests, gradient-boosted trees, isotonic regression

More details for these methods can be found here:

- [Linear models](#)
 - [binary classification \(SVMs, logistic regression\)](#)
 - [linear regression \(least squares, Lasso, ridge\)](#)
- [Decision trees](#)
- [Ensembles of decision trees](#)
 - [random forests](#)
 - [gradient-boosted trees](#)
- [Naive Bayes](#)
- [Isotonic regression](#)




MLlib

Classification: RF

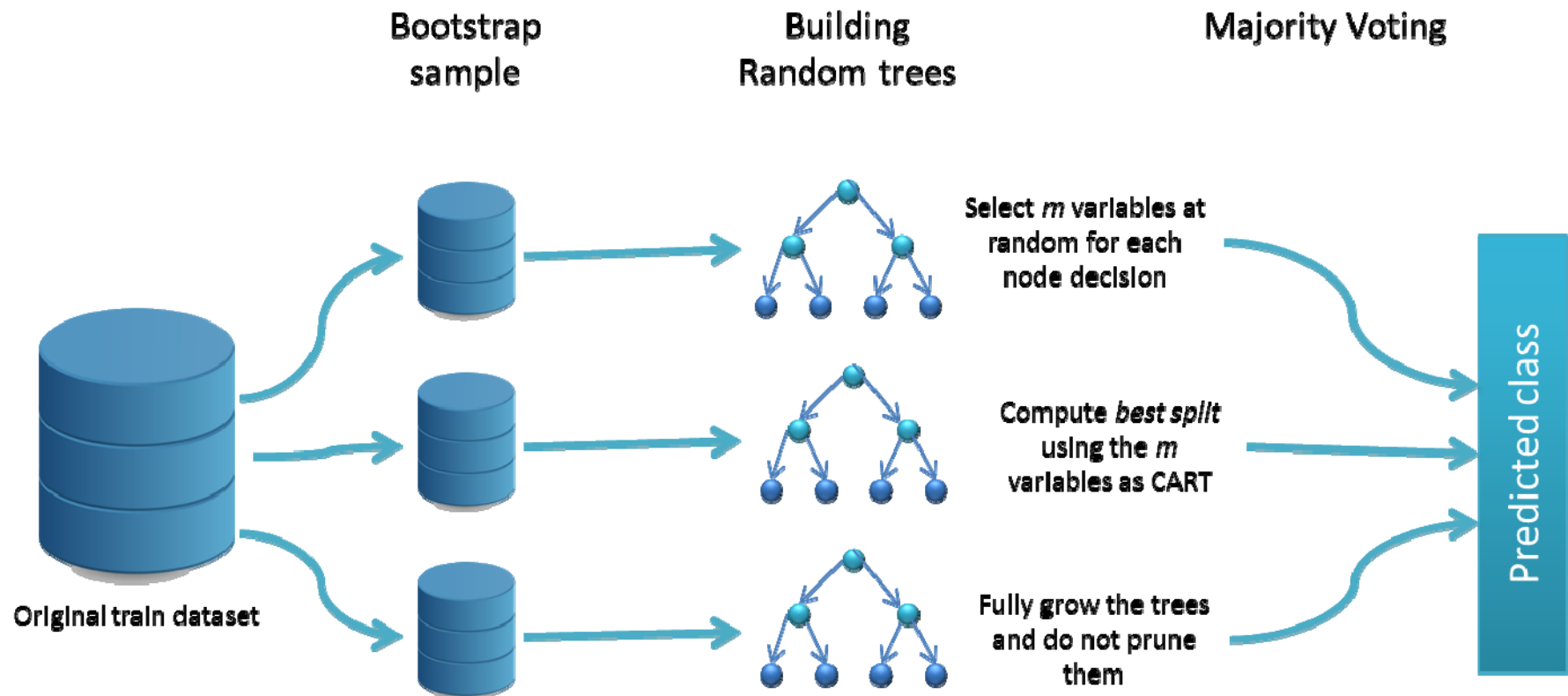


Scalable machine learning and data mining



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


Case of Study: Random Forest for KddCup'99



Classification: RF



Scalable machine learning and data mining

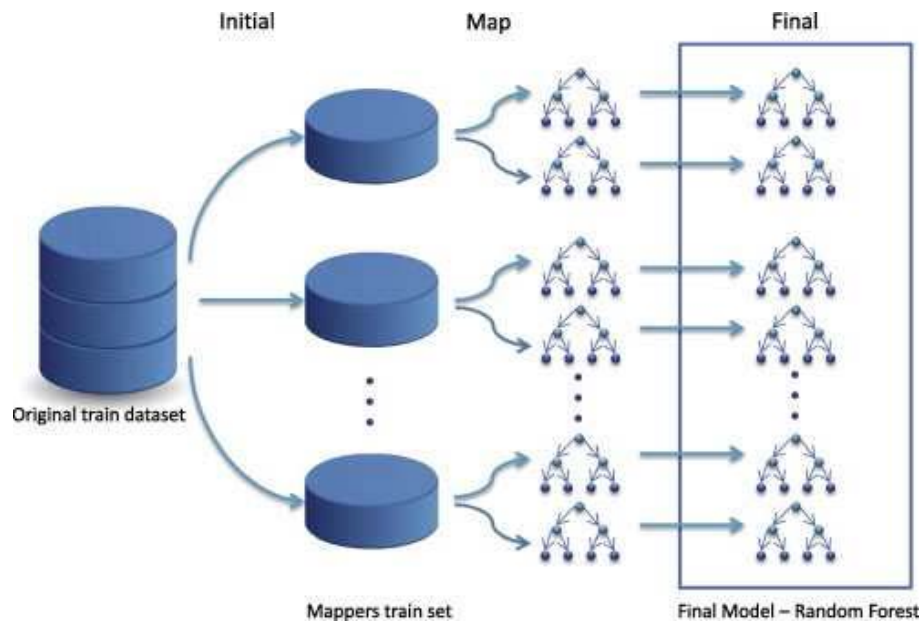


Apache Mahout has implementations of a wide range of machine learning and data mining algorithms: clustering, classification, collaborative filtering and frequent pattern mining

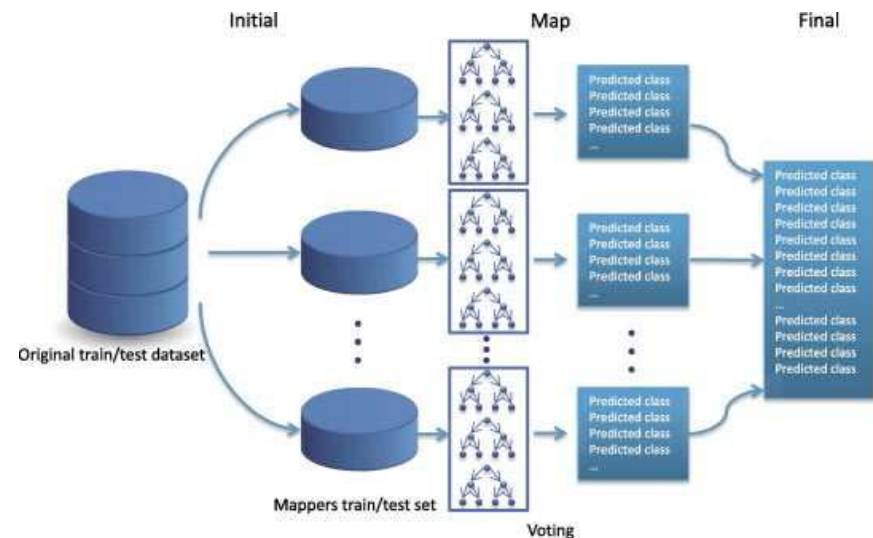
Case of Study: Random Forest for KddCup'99

The **RF Mahout Partial** implementation: is an algorithm that builds multiple trees for different portions of the data. Two phases:

Building phase




Classification phase



Classification: RF



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

Case of Study: Random Forest for KddCup'99

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


Time elapsed (seconds) for sequential versions:

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

Classification: RF



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

Case of Study: Random Forest for KddCup'99

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

Time elapsed (seconds) for Big data versions with 20 partitions:

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

Classification: Data Preprocessing

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 parameters in the constructor:

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

Classification: Data Preprocessing



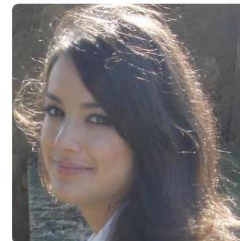
Our approaches:



Evolutionary
data reduction

<https://github.com/triguero>

Isaac Triguero



Sara Del Río

Preprocessing
imbalance classification

<https://github.com/saradelrio>

hadoop-imbalanced-preprocessing

MapReduce implementations of random oversampling, random undersampling ;

(SMOTE) algorithms using Hadoop

Popular repositories

MR-EFS

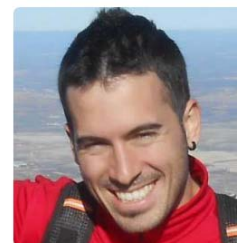
This project includes the implementation of evolutionary feature s

MRPR

This repository includes the MapReduce implementation propose

ROSEFW-RF

This project contains the code used in the ROSEFW-RF paper.



Sergio Ramírez

<https://github.com/sramirez>

Feature selection
and discretization

fast-mRMR

An improved implementation of the classical f...

spark-infotheoretic-feature-sel...

This package contains a generic implementati...

spark-MDLP-discretization

Spark implementation of Fayyad's discretizer...

Spark Packages

[Feedback](#) [Register a package](#) [Login](#)

spark-MDLP-discretization ([homepage](#))

Spark implementation of Fayyad's discretizer based on Minimum Description Length Principle (MDLP)

@sramirez / ★★★★★ (14)

spark-infotheoretic-feature-selection ([homepage](#))

Feature Selection framework based on Information Theory that includes: mRMR, InfoGain, JMI and other commonly used F

@sramirez / ★★★★★ (14)

Classification: Fuzzy Rule Based Systems



Our approaches:

Fuzzy Rule Based System for classification

Fuzzy Rule Based System with cost sensitive for imbalanced data sets



Sara Del Río

<https://github.com/saradelrio>

Popular repositories

[Chi-FRBCS-BigData-Ave](#)

Chi-FRBCS-BigData-Ave: MapReduce implementation of the Chi et al.'s approach.

[Chi-FRBCS-BigData-Max](#)

Chi-FRBCS-BigData-Max: MapReduce implementation of the Chi et al.'s approach.

[Chi-FRBCS-BigDataCS](#)

Chi-FRBCS-BigDataCS: MapReduce implementation of the basic Chi et al.'s algorithm

[hadoop-imbalanced-preprocessi](#)

[ng](#)

MapReduce implementations of random oversampling, random undersampling and "Synthetic Minority Over-sampling Technique" (SMOTE) algorithms using Hadoop

[RF-BigDataCS](#)

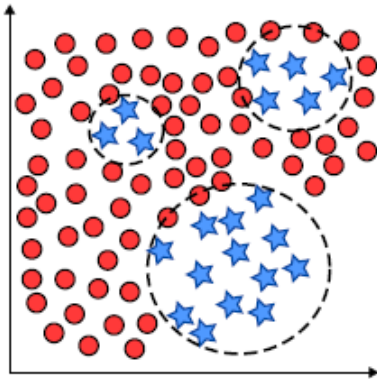
RF-BigDataCS: A cost-sensitive approach for Random Forest MapReduce algorithm to

Big Data Classification and Preprocessing

Tasks to discuss:

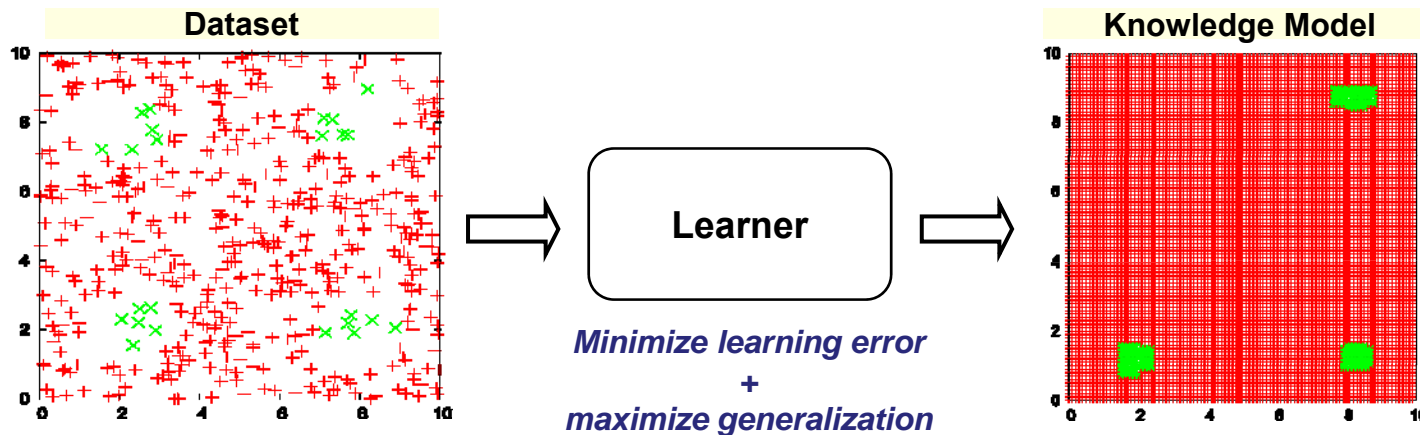
1. Scalability of the proposals (Algorithms redesign!!)
2. Reduce phase: How must we combine the output of the maps? (Fundamental phase to use MapReduce for Big Data Preprocessing!!)
3. Appearance of small disjuncts with the MapReduce data fragmentation.
This problem is basically associated to imbalanced classification: Lack of Data/lack of density between classes

Appearance of small disjuncts with the MapReduce data fragmentation



(b) Small disjuncts

Rare cases or Small disjuncts are those disjuncts in the learned classifier that cover few training examples.

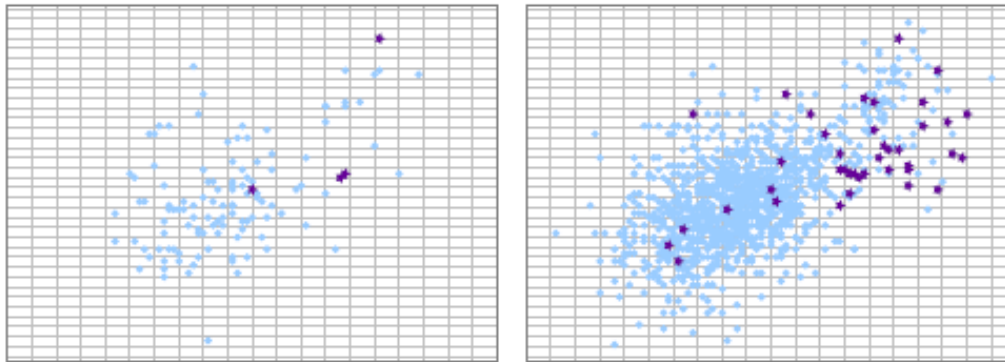


T. Jo, N. Japkowicz. Class imbalances versus small disjuncts. SIGKDD Explorations 6:1 (2004) 40-49

G.M. Weiss. Mining with Rarity: A Unifying Framework. SIGKDD Explorations 6:1 (2004) 7-19

Appearance of small disjuncts with the MapReduce data fragmentation

Data fragmentation - Lack of data



(a) 10 % of training instances

(b) 100 % of training instances

Figure 11: Lack of density or small sample size on the yeast4 dataset

The lack of data in the training data may also cause the introduction of small disjuncts.

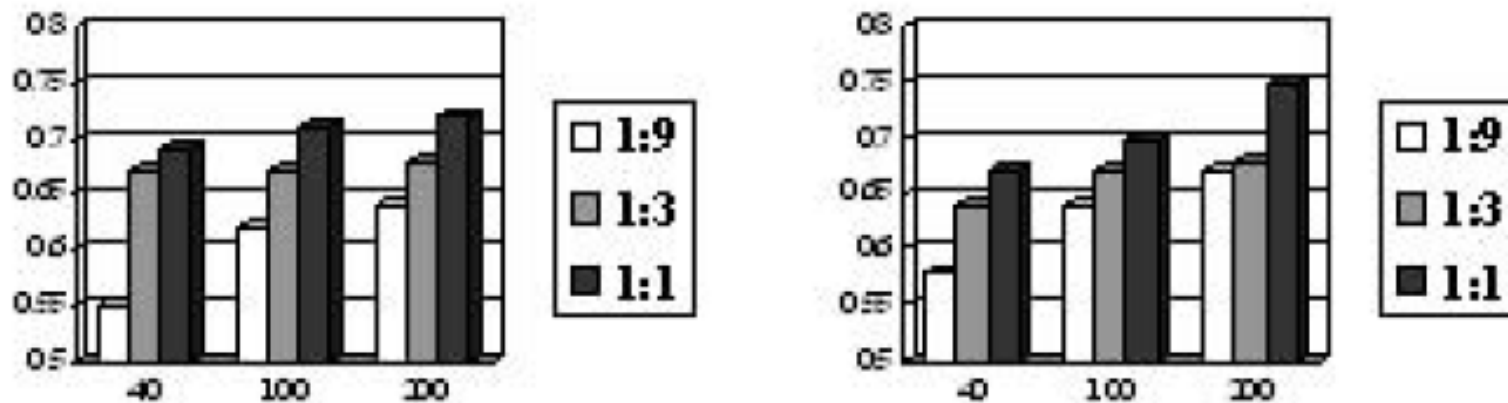
It becomes very hard for the learning algorithm to obtain a model that is able to perform a good generalization when there is not enough data that represents the boundaries of the problem.

And, what it is also most significant, when the concentration of minority examples is so low that they can be simply treated as noise.

Appearance of small disjuncts with the MapReduce data fragmentation

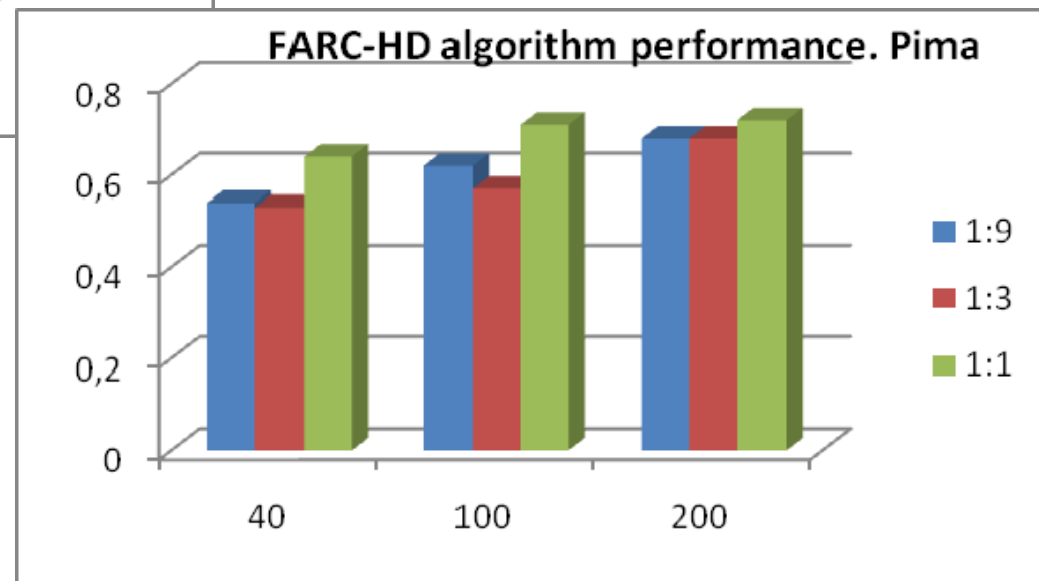
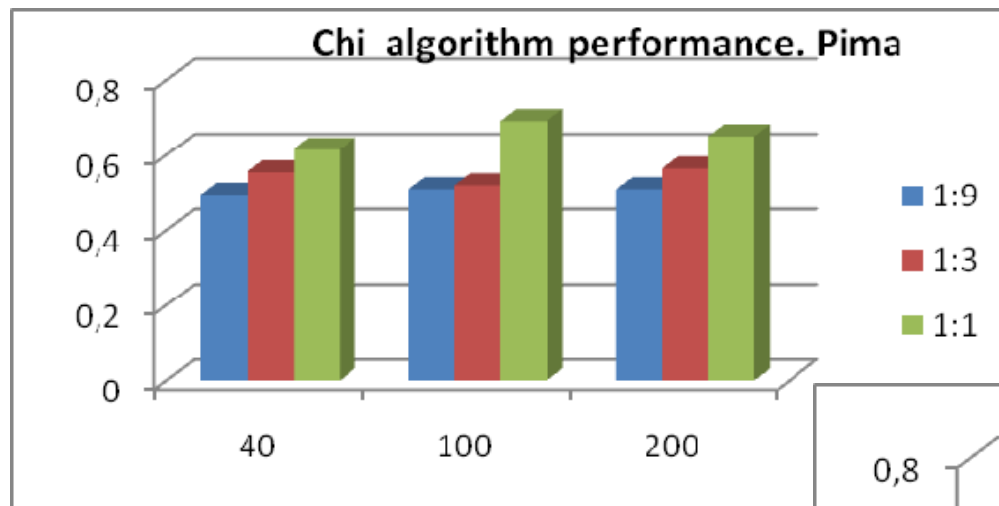
Lack of data

Left-C4.5, right-Backpropagation (Pima and Wisconsin Breast Cancer): These results show that the performance of classifiers, though hindered by class imbalances, is repaired as the training set size increases. This suggests that small disjuncts play a role in the performance loss of class imbalanced domains.



Appearance of small disjuncts with the MapReduce data fragmentation

Lack of data. Fuzzy models performance



Robustness to the lack of data?



Outline

IFSA – EUSFLAT 2015
Celebrating fifty years of Fuzzy Sets

- ❑ Big Data. Big Data Science
- ❑ Why Big Data? MapReduce Paradigm. Hadoop Ecosystem
- ❑ Big Data Classification: Learning algorithms
- ❑ Big Data Classification: Computational Intelligence Approches
- ❑ Big Data Classification: Imbalanced classes
- ❑ Final Comments

Big Data: Selected Computational Intelligence approaches



- **Chi-FRBCS-BigData algorithm:** A MapReduce Design based on the Fusion of **Fuzzy Linguistic Rules** for **classification**
- **MRPR:** A Combined MapReduce-Windowing Two-Level Parallel Scheme for **Evolutionary Prototype Generation**
- **MR-EFS: Evolutionary Feature Selection** for Big Data Classification: A MapReduce Approach
- **Chi-FRBCS-BigDataCS algorithm** for **imbalanced bigdata classification**: A MapReduce design
- **Evolutionary Feature Weighting:** ROSEFW-RF algorithm for ECBDL'14 Big Data Competition. **Imbalanced Big Data classification**

Big Data: Selected Computational Intelligence approaches



- **Chi-FRBCS-BigData algorithm: A MapReduce Design based on the Fusion of Fuzzy Linguistic Rules**
- MRPR: A Combined MapReduce-Windowing Two-Level Parallel Scheme for Evolutionary Prototype Generation
- MR-EFS: Evolutionary Feature Selection for Big Data Classification: A MapReduce Approach
- Chi-FRBCS cost sensitive algorithm for imbalanced big data: A MapReduce design
- **Evolutionary Feature Weighting: ROSEFW-RF algorithm for ECBDL'14 Big Data Competition. (Imbalanced Big Data classification)**

Uncertainty and Big Data

- Uncertainty is inherent to Big Data due to
 - Heterogeneous sources
 - Variety in data
 - Incomplete data
 - Veracity in question
- Fuzzy Rule Based Classification Systems can manage
 - Uncertainty
 - Vagueness
 - Lack of data



Big Data: Selected Computational Intelligence approaches



Chi-FRBCS-BigData: A Case of Study

We choose a simple Learning Methods to analyze the potential of FRBCSs for Big Data Classification

- MapReduce design based on the FRBCS algorithm (Chi et al).
- Uses two different MapReduce processes
 - Phase 1: Building the Fuzzy Rule Base
 - Phase 2: Estimating the class of samples belonging to big data sample sets
- Two versions which differ in the *Reduce function* of the building of the FRB have been produced
 - Chi-FRBCS-BigData-Max
 - Chi-FRBCS-BigData-Average

S. Río, V. López, J.M. Benítez, F. Herrera. *A MapReduce Approach to Address Big Data Classification Problems Based on the Fusion of Linguistic Fuzzy Rules*. International Journal of Computational Intelligence Systems 8:3 (2015) 422-437. doi: [10.1080/18756891.2015.1017377](https://doi.org/10.1080/18756891.2015.1017377)

Big Data: Selected Computational Intelligence approaches



Chi-FRBCS

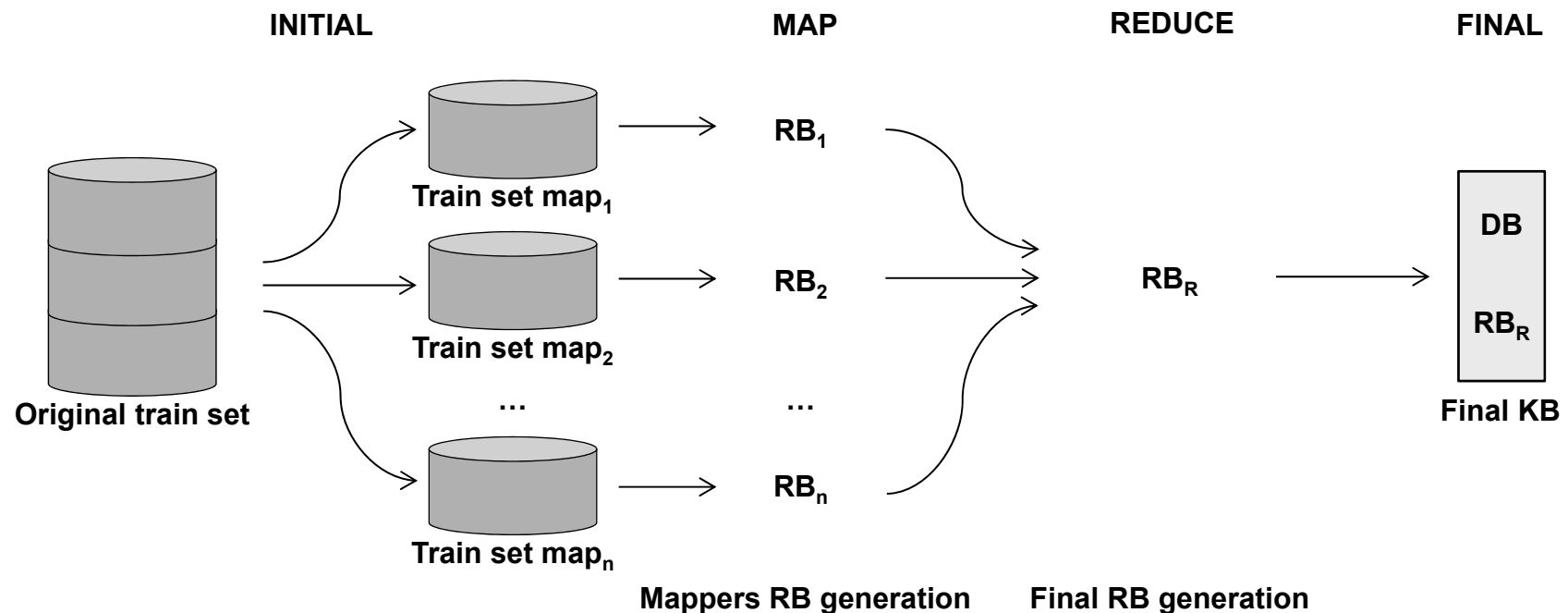
- Produces rules like “**Rule R_j** : IF x_1 IS A_j^1 AND ... AND x_n IS A_j^n THEN Class = C_j with RW_j ”
- Builds the fuzzy partition using equally distributed triangular membership functions
- Builds the RB creating a fuzzy rule associated to each example
- Rules with the same antecedent may be created:
 - Same consequent → Delete duplicated rules
 - Different consequent → Preserve highest weight rule

Z. Chi, H. Yan and T. Pham, Fuzzy algorithms with applications to image processing and pattern recognition, World Scientific, 1996.

Big Data: Selected Computational Intelligence approaches



Building the RB with Chi-FRBCS-BigData: A Map Reduce approach



The key of a MapReduce data partitioning approach is usually on the reduce phase

Two alternative reducers (Max vs average weights)

Big Data: Selected Computational Intelligence approaches



Building the FRB with Chi-FRBCS-BigData-Max

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₁ = 0.8743
 R₂: IF A₁ = L₂ AND A₂ = L₂ THEN C₂; RW₂ = 0.9142
 ...

RB₁

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₂; RW₃ = 0.9254
 R₂: IF A₁ = L₁ AND A₂ = L₂ THEN C₂; RW₂ = 0.8842
 ...

RB₂

R₁: IF A₁ = L₂ AND A₂ = L₁ THEN C₂; RW₃ = 0.6534
 R₂: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₁ = 0.7142
 ...

RB₃

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₂; RW₁ = 0.2143
 R₂: IF A₁ = L₃ AND A₂ = L₂ THEN C₂; RW₃ = 0.4715
 ...

RB₄

R₁: IF A₁ = L₂ AND A₂ = L₃ THEN C₂; RW₃ = 0.7784
 R₂: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₂ = 0.8215
 ...

RB_n

REDUCE

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₁ = 0.9254
 R₂: IF A₁ = L₂ AND A₂ = L₂ THEN C₂; RW₂ = 0.9142
 R₃: IF A₁ = L₁ AND A₂ = L₂ THEN C₂; RW₂ = 0.8842
 R₄: IF A₁ = L₂ AND A₂ = L₁ THEN C₂; RW₃ = 0.6534
 R₅: IF A₁ = L₃ AND A₂ = L₂ THEN C₂; RW₃ = 0.4715
 R₆: IF A₁ = L₂ AND A₂ = L₃ THEN C₂; RW₃ = 0.7784
 ...

RB_R

Final RB generation

RB₁, R₁, C₁, RW = 0.8743
RB₂, R₁, C₂, RW = 0.9254
 RB₃, R₂, C₁, RW = 0.7142
 RB₄, R₁, C₂, RW = 0.2143
 RB₅, R₂, C₁, RW = 0.8215

Big Data: Selected Computational Intelligence approaches



Building the FRB with Chi-FRBCS-BigData-Ave

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₁ = 0.8743
 R₂: IF A₁ = L₂ AND A₂ = L₂ THEN C₂; RW₂ = 0.9142
 ...

RB₁

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₂; RW₃ = 0.9254
 R₂: IF A₁ = L₁ AND A₂ = L₂ THEN C₂; RW₂ = 0.8842
 ...

RB₂

R₁: IF A₁ = L₂ AND A₂ = L₁ THEN C₂; RW₃ = 0.6534
 R₂: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₁ = 0.7142
 ...

RB₃

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₂; RW₁ = 0.2143
 R₂: IF A₁ = L₃ AND A₂ = L₂ THEN C₂; RW₃ = 0.4715
 ...

RB₄

...

R₁: IF A₁ = L₂ AND A₂ = L₃ THEN C₂; RW₃ = 0.7784
 R₂: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₂ = 0.8215
 ...

RB_n

REDUCE

R₁: IF A₁ = L₁ AND A₂ = L₁ THEN C₁; RW₁ = 0.8033
 R₂: IF A₁ = L₂ AND A₂ = L₂ THEN C₂; RW₂ = 0.9142
 R₃: IF A₁ = L₁ AND A₂ = L₂ THEN C₂; RW₂ = 0.8842
 R₄: IF A₁ = L₂ AND A₂ = L₁ THEN C₂; RW₃ = 0.6534
 R₅: IF A₁ = L₃ AND A₂ = L₂ THEN C₂; RW₃ = 0.4715
 R₆: IF A₁ = L₂ AND A₂ = L₃ THEN C₂; RW₃ = 0.7784
 ...

RB_R

Final RB generation

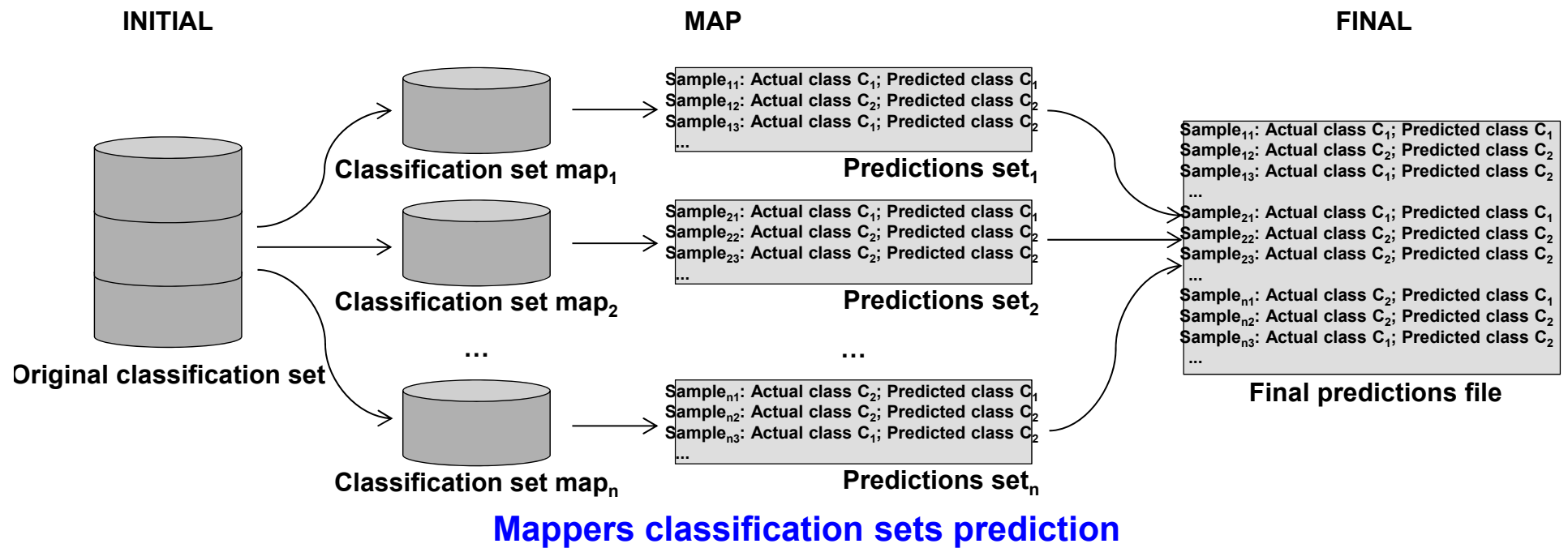
RB₁, R₁, C₁, RW = 0.8743
 RB₂, R₁, C₂, RW = 0.9254
 RB₃, R₂, C₁, RW = 0.7142
 RB₄, R₁, C₂, RW = 0.2143
 RB₅, R₂, C₁, RW = 0.8215

RC₁, C₁, RW_{ave} = 0.8033
 RC₂, C₂, RW_{ave} = 0.5699

Big Data: Selected Computational Intelligence approaches



Estimating the class of a Big dataset with Chi-FRBCS-BigData



Big Data: Selected Computational Intelligence approaches



Experimental Analysis: Chi-FRBCS-BigData

- 6 Datasets with two classes problem
- Stratified 10 fold cross-validation
- Parameters:
 - Conjunction Operator: Product T-norm
 - Rule Weight: Penalized Certainty Factor
 - Fuzzy Reasoning Method: Winning Rule
 - Number of fuzzy labels per variable: 3 labels
 - Number of mappers: 16, 32, 64

Experimental Framework

Datasets	#Ex.	#Atts.	Selected classes	#Samples per class
RLCP	5749132	2	(FALSE; TRUE)	(5728201; 20931)
<u>Kddcup_DOS_vs_normal</u>	4856151	41	(DOS; normal)	(3883370; 972781)
<u>Poker_o_vs_1</u>	946799	10	(0; 1)	(513702; 433097)
<u>Covtype_2_vs_1</u>	495141	54	(2; 1)	(283301; 211840)
<u>Census</u>	141544	41	(-50000.; 50000+.)	(133430; 8114)
<u>Fars Fatal Inj vs No Inj</u>	62123	29	(Fatal Inj; No Inj)	(42116; 20007)

Big Data: Selected Computational Intelligence approaches



Analysis of the Performance, Precision

Datasets	8 maps					
	Chi-FRBCS		Chi-BigData-Max		Chi-BigData-Ave	
	Acc_{tr}	Acc_{tst}	Acc_{tr}	Acc_{tst}	Acc_{tr}	Acc_{tst}
Poker_0_vs_1	63.72	61.77	62.93	60.74	63.12	60.91
Covtype_2_vs_1	74.65	74.57	74.69	74.63	74.66	74.61
Census	96.52	86.06	97.12	93.89	97.12	93.86
Fars_Fatal_Inj_vs_No_Inj	99.66	89.26	97.01	95.07	97.18	95.25
Average	83.64	77.92	82.94	81.08	83.02	81.16

Good precision!

Big Data: Selected Computational Intelligence approaches



Analysis of the Performance, Number of rules

Kddcup_DOS_vs_normal dataset

NumRules by map

Final numRules

RB_1 size: 211

RB_R size: 301

RB_2 size: 212

RB_3 size: 221

RB_4 size: 216

RB_5 size: 213

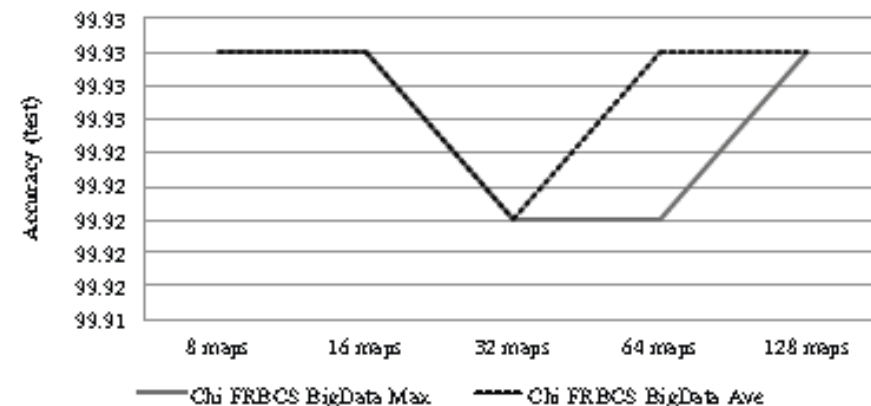
RB_6 size: 210

RB_7 size: 211

RB_8 size: 214

Robustness to the lack of data increasing the final number of rules

Class	Instance Number
normal	972.781
DOS	3.883.370



(b) Kddcup_DOS_vs_normal dataset

Big Data: Selected Computational Intelligence approaches



Analysis of the Performance, Number of rules

Datasets	8 maps		
	Chi-FRBCS Average NumRules	Chi-BigData-Max Average NumRules	Chi-BigData-Ave Average NumRules
Census	31518.3	34278.0	34278.0
Covtype_2_vs_1	6962.7	7079.1	7079.1
Fars_Fatal_Inj_vs_No_Inj	16843.3	17114.9	17114.9
Poker_0_vs_1	51265.4	52798.1	52798.1

Robustness to the lack of data for the data fragmentation, increasing the final number of rules

This may cause a improvement in the performance

Big Data: Selected Computational Intelligence approaches



Analysis of the Performance, Precision

Datasets	16 mappers			
	Chi-BigData-Max		Chi-BigData-Ave	
	Acc _{tr}	Acc _{tst}	Acc _{tr}	Acc _{tst}
RLCP	99.63	99.63	99.63	99.63
Kddcup_DOS_vs_normal	99.93	99.93	99.93	99.93
Poker_o_vs_1	62.18	59.88	62.58	60.35
Covtype_2_vs_1	74.77	74.72	74.77	74.69
Census	97.14	93.75	97.15	93.52
Fars_Fatal_Inj_vs_No_Inj	96.69	94.75	97.06	95.01
Average	88.39	87.11	88.52	87.19

Datasets	32 mappers			
	Chi-BigData-Max		Chi-BigData-Ave	
	Acc _{tr}	Acc _{tst}	Acc _{tr}	Acc _{tst}
RLCP	99.63	99.63	99.63	99.63
Kddcup_DOS_vs_normal	99.92	99.92	99.92	99.92
Poker_o_vs_1	61.27	58.93	61.82	59.30
Covtype_2_vs_1	74.69	74.62	74.88	74.85
Census	97.11	93.48	97.12	93.32
Fars_Fatal_Inj_vs_No_Inj	96.49	94.26	96.87	94.63
Average	88.9	86.81	88.37	86.94

Datasets	64 mappers			
	Chi-BigData-Max		Chi-BigData-Ave	
	Acc _{tr}	Acc _{tst}	Acc _{tr}	Acc _{tst}
RLCP	99.63	99.63	99.63	99.63
Kddcup_DOS_vs_normal	99.92	99.92	99.93	99.93
Poker_o_vs_1	60.45	57.95	60.88	58.12
Covtype_2_vs_1	74.67	74.52	75.05	74.96
Census	97.07	93.30	97.13	93.11
Fars_Fatal_Inj_vs_No_Inj	96.27	93.98	96.76	94.56
Average	88.00	86.55	88.23	86.72

- Performance improves slightly with less maps (alleviate the small sample size problem)
- Chi-BigData-Ave obtains slightly better classification results

Big Data: Selected Computational Intelligence approaches



Analysis of the Performance, Runtime (Chi-BigData-Ave)

Datasets	8 maps		
	Chi-FRBCS Runtime (s)	Chi-BigData-Max Runtime (s)	Chi-BigData-Ave Runtime (s)
Census	38655.60	1102.45	1343.92
Covtype_2_vs_1	86247.70	2482.09	2512.16
Fars_Fatal_Inj_vs_No_Inj	8056.60	241.96	311.95
Poker_0_vs_1	114355.80	5672.80	7682.19
Average	61828.93	2374.82	2962.56

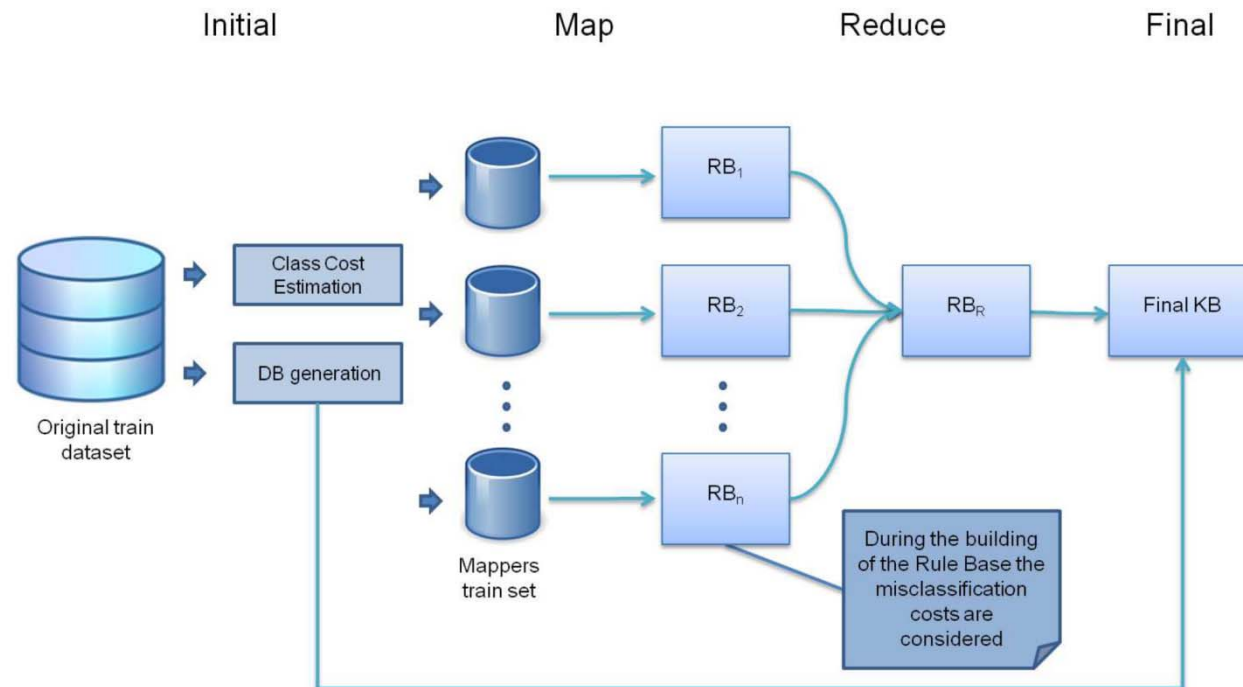
KddCUP'99		Maps number	Seconds
Class	Instance Number	8	116.218,26
normal	972.781	16	29.820,01
DOS	3.883.370	32	7.708,96
		64	2.096,34
		132	1.579,77

Big Data: Selected Computational Intelligence approaches



FRBCS for Big Data: Model for Imbalanced classes

- Chi-FRBCS-BigDataCS: **algorithm for imbalanced bigdata**



V. López, S. Río, J.M. Benítez, F. Herrera. *Cost-Sensitive Linguistic Fuzzy Rule Based Classification Systems under the MapReduce Framework for Imbalanced Big Data*. *Fuzzy Sets and Systems* 258 (2015) 5-38.

Big Data: Selected Computational Intelligence approaches



FRBCS for Big Data: Final Comments

- **Linguistic FRBCS for Big Data** (Chi-FRBCS-BigData) under the MapReduce framework:
 - Manages big datasets
 - Without damaging the classification accuracy
 - Fast response times (increasing with the number of Maps)
- **It is a promising line of work for the design of high performance Fuzzy Models for Big Data**



Outline

IFSA – EUSFLAT 2015
Celebrating fifty years of Fuzzy Sets

- ❑ Big Data. Big Data Science
- ❑ Why Big Data? MapReduce Paradigm. Hadoop Ecosystem
- ❑ Big Data Classification: Learning algorithms
- ❑ Big Data Classification: Computational Intelligence Approches
- ❑ **Big Data Classification: Imbalanced classes**
- ❑ Final Comments

Evolutionary Computation for Big Data and Big Learning Workshop

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

Objective: Contact map prediction

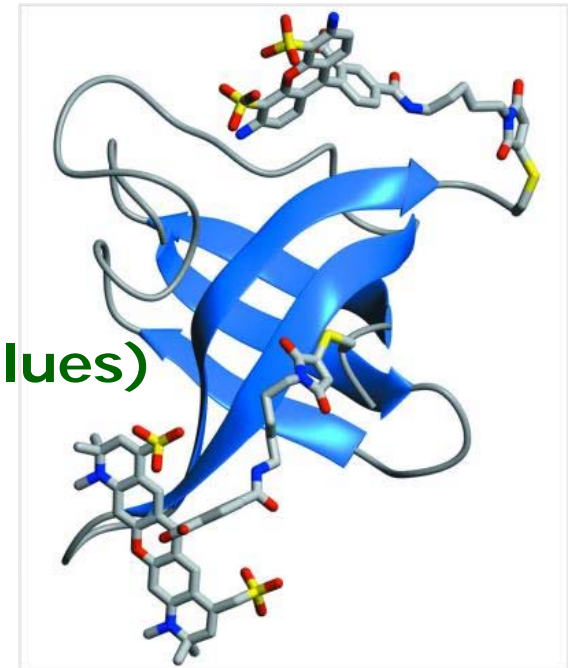
Details:

- ❑ 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>



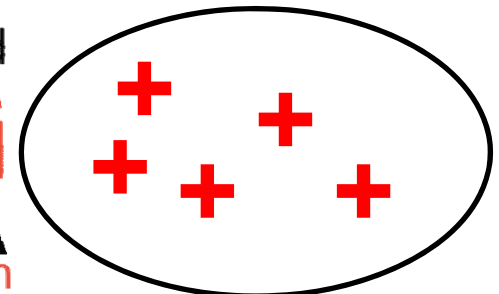
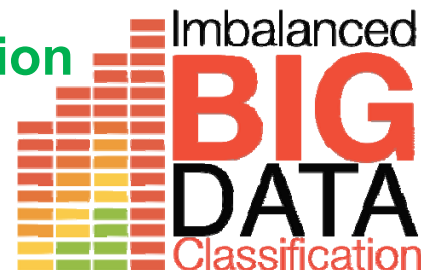
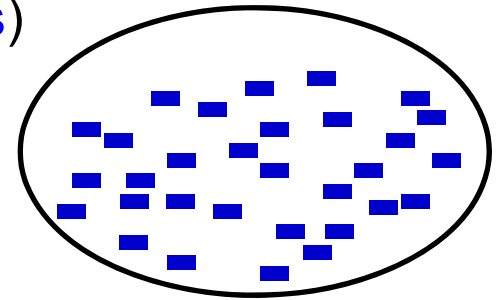
J. Bacardit et al, Contact map prediction using a large-scale ensemble of rule sets and the fusion of multiple predicted structural features, *Bioinformatics* 28 (19) (2012) 2441-2448

Evolutionary Computation for Big Data and Big Learning Workshop

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

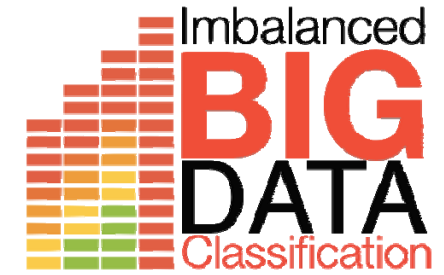
The challenge:

- ❑ 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

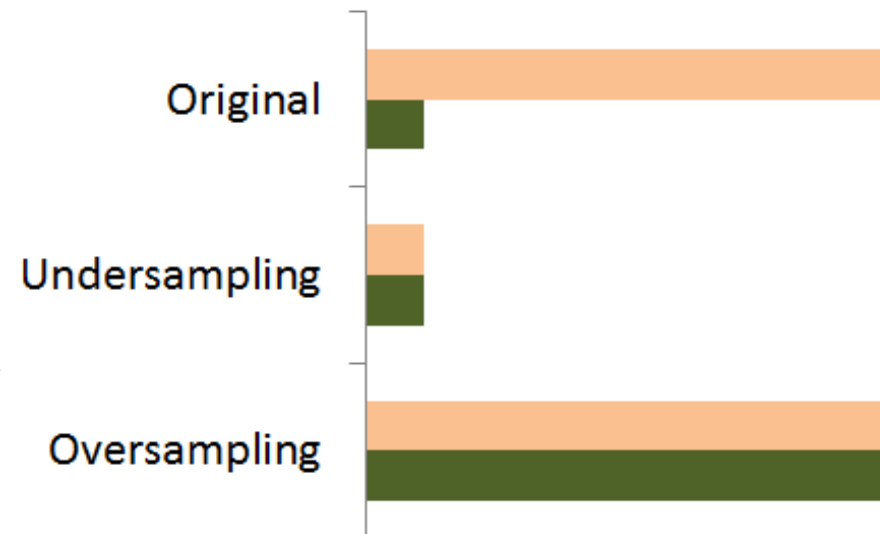
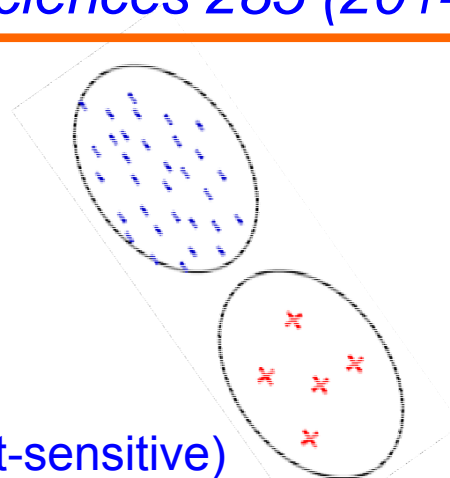
Under-Sampling

Random

Focused

Cost Modifying (cost-sensitive)

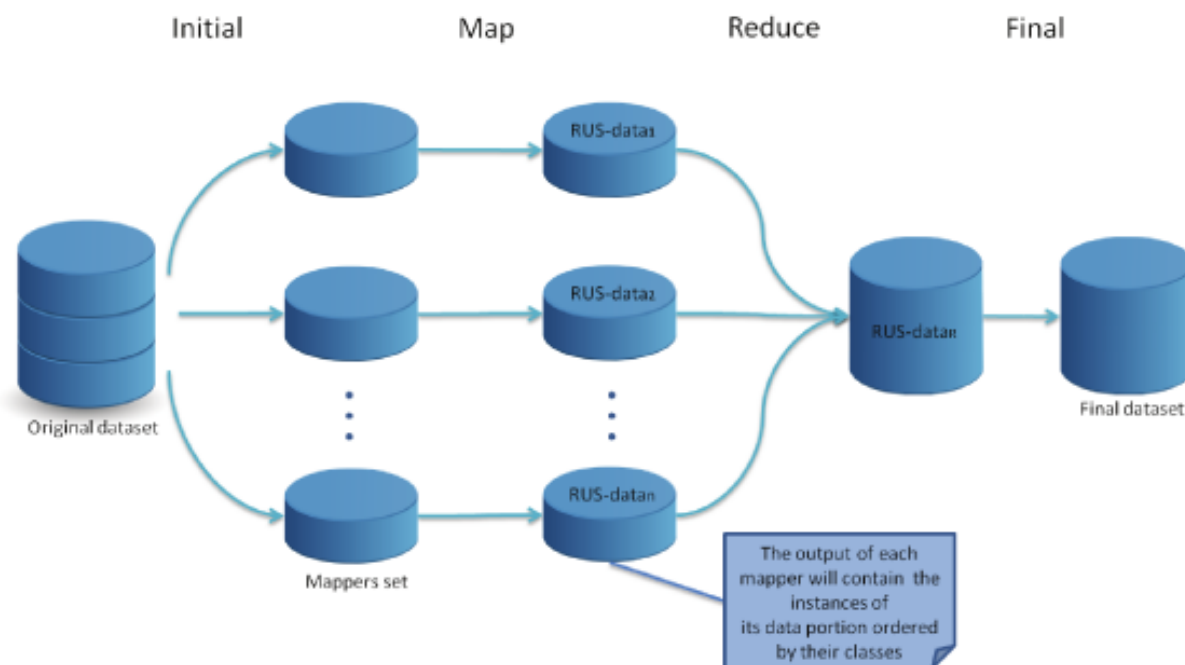
Boosting/Bagging approaches (with preprocessing)



Imbalanced Big Data Classification

A MapReduce Approach for Random Undersampling

Random undersampling for big data (RUS+RF-BigData):

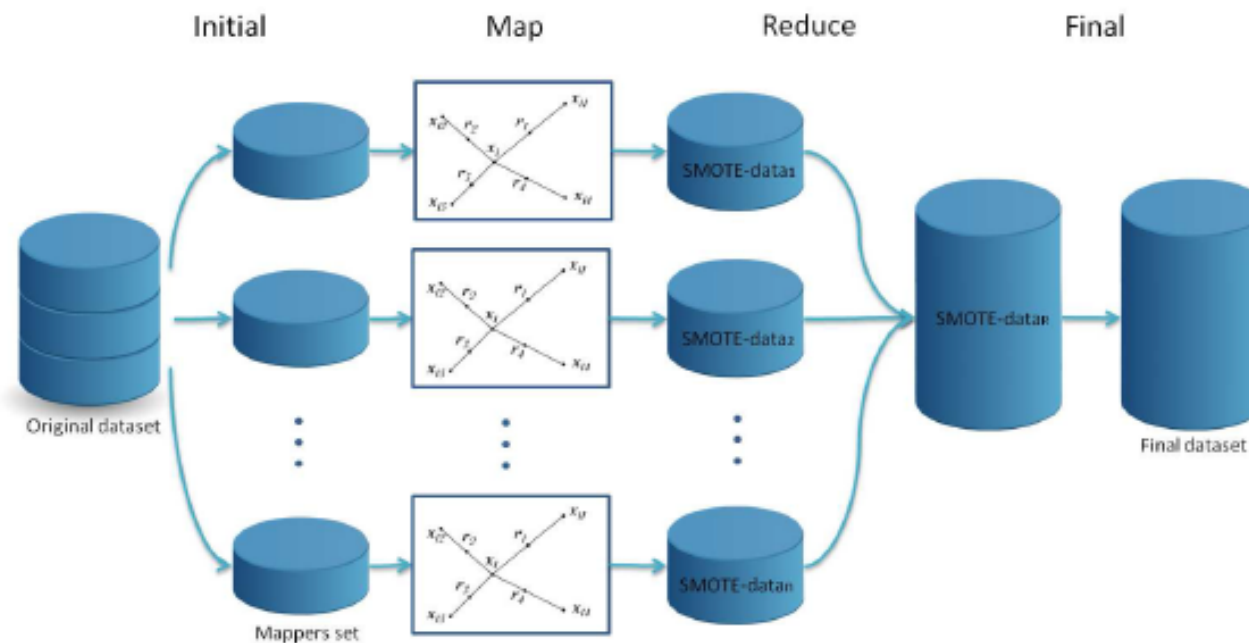


S. Río, V. López, J.M. Benítez, F. Herrera, On the use of MapReduce for Imbalanced Big Data using Random Forest. *Information Sciences* 285 (2014) 112-137.

Imbalanced Big Data Classification

A MapReduce Approach for Random Oversampling

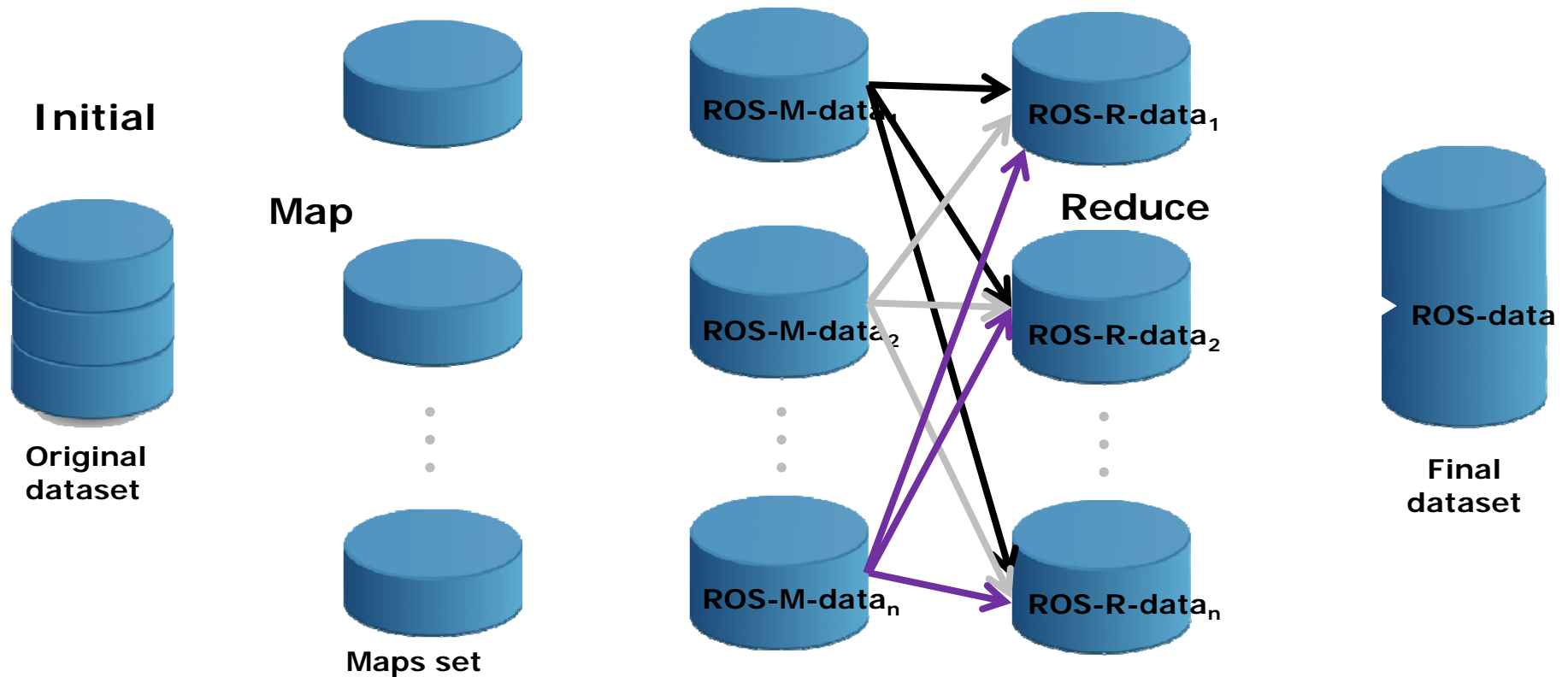
SMOTE for big data (SMOTE+RF-BigData):



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.

Imbalanced Big Data Classification

A MapReduce Approach for Random Oversampling



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.

Imbalanced Big Data Classification

Dataset	Average (kddcup)							
	8 mappers		16 mappers		32 mappers		64 mappers	
	GM _{tr}	GM _{tst}	GM _{tr}	GM _{tst}	GM _{tr}	GM _{tst}	GM _{tr}	GM _{tst}
Big data versions								
RF-BigData	0.7620	0.7505	0.6985	0.6976	0.6852	0.6836	0.6626	0.6598
RF-BigDataCS	0.9404	0.9305	0.9480	0.9651	0.9173	0.9328	0.9372	0.9286
ROS+RF-BigData	1.0000	0.9661	0.9999	0.9696	0.9999	0.9773	0.9999	0.9857
RUS+RF-BigData	0.9869	0.9843	0.9490	0.9336	0.7103	0.7104	0.7049	0.7048
SMOTE+RF-BigData	0.9477	0.9140	0.9381	0.9191	0.9445	0.9091	0.8994	0.8722

Analysis of the effectiveness in classification of the approaches

(Potential problem: lack of density of the positive class for RUS/SMOTE)

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.

Evolutionary Computation for Big Data and Big Learning Workshop

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

Objective: Contact map prediction

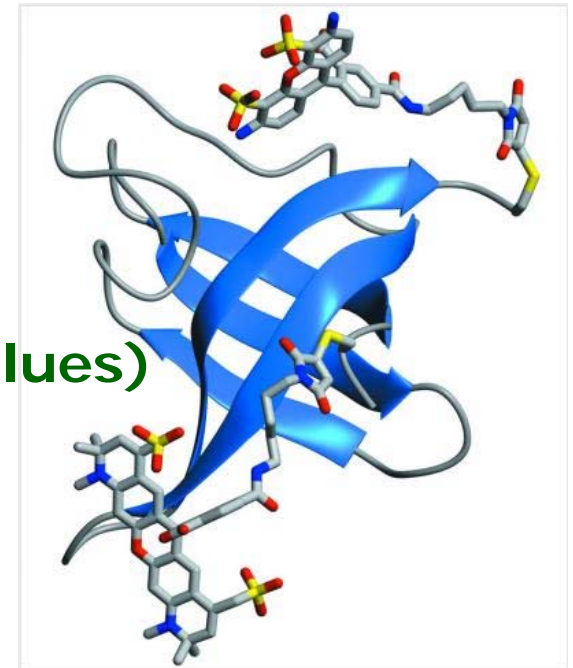
Details:

- ❑ 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

ECBDL'14 Big Data Competition

ECBDL'14 Big Data Competition 2014

Our approach:

1. Balance the original training data

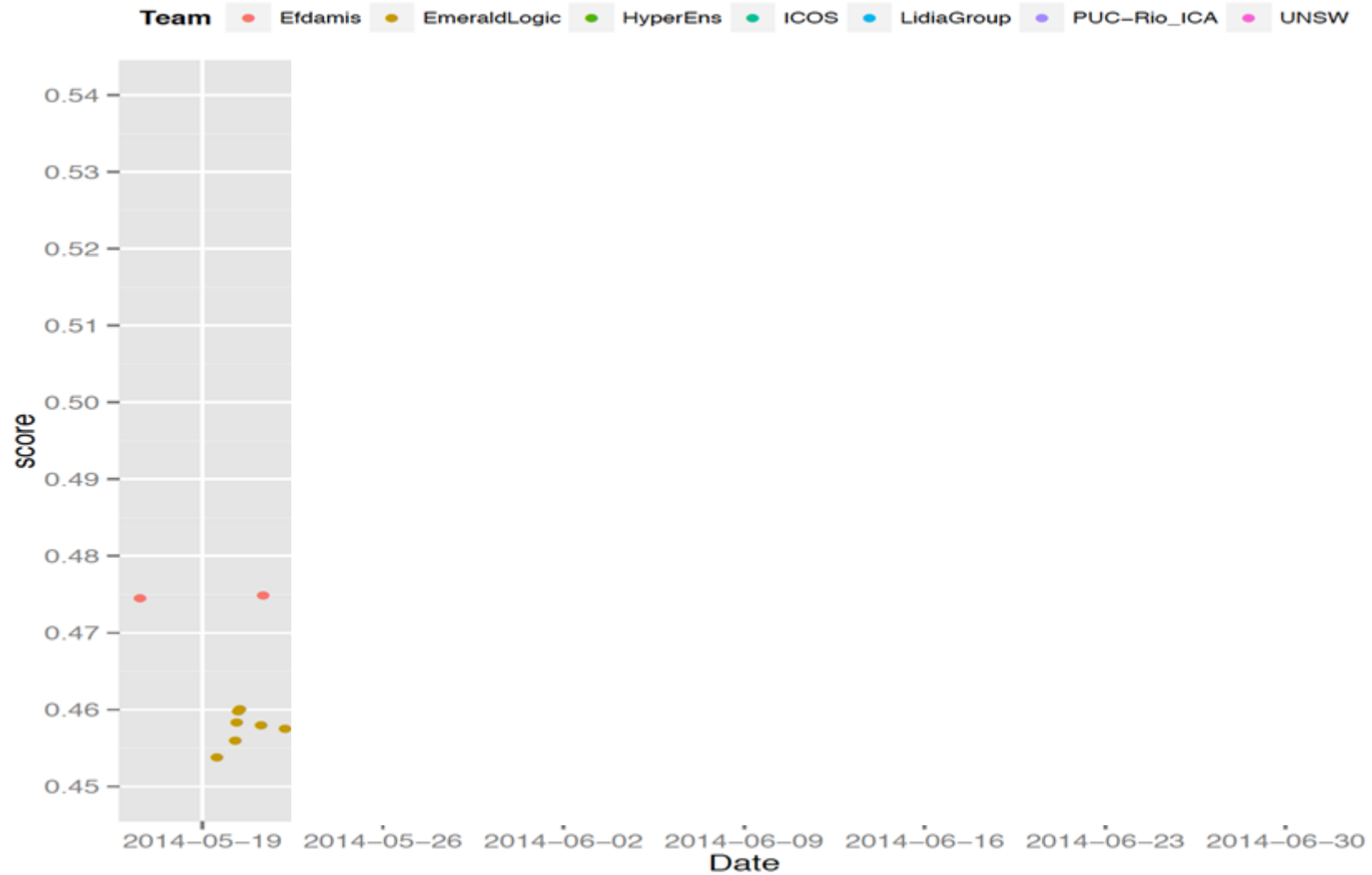
- ❑ Random Oversampling
- ❑ (As first idea, it was extended)

2. Learning a model.

- ❑ Random Forest



ECBDL'14 Big Data Competition



ECBDL'14 Big Data Competition

We initially focused on

❑ Oversampling rate: { 100% }

RandomForest:

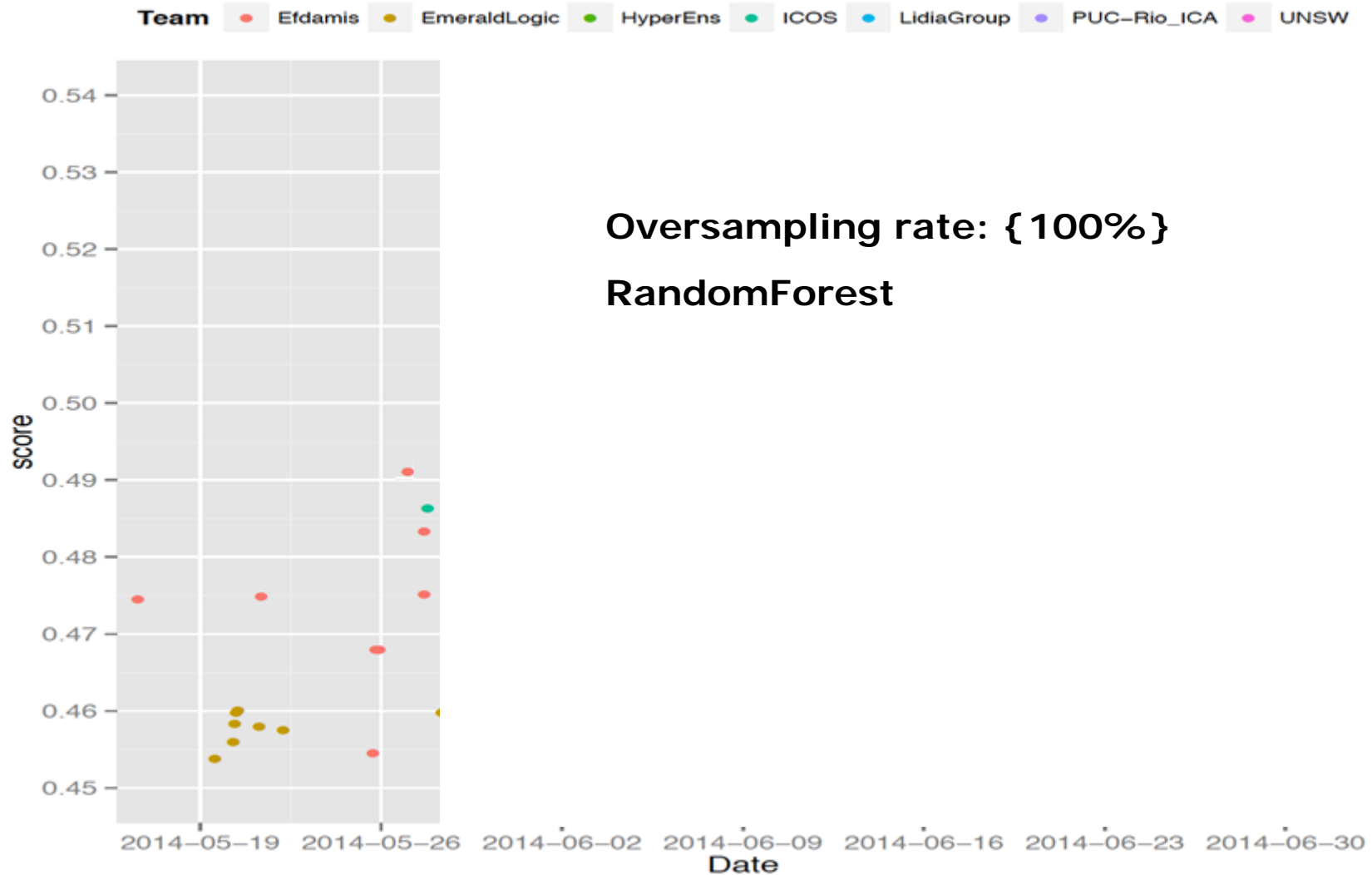
❑ Number of used features: 10 ($\log n + 1$); Number of trees: 100

❑ Number of maps: { 64, 190, 1024, 2048 }

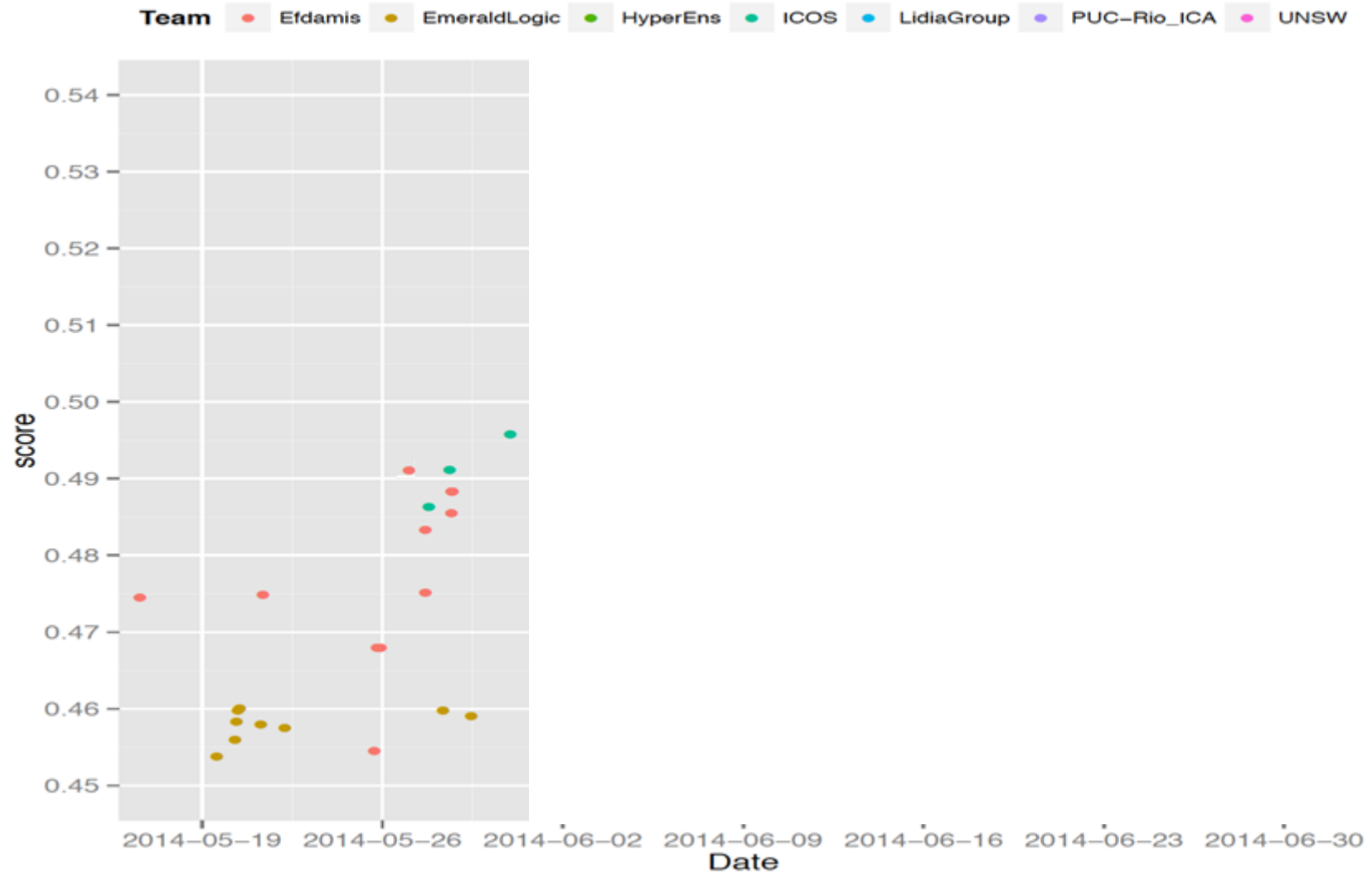
Nº mappers	TPR_tst	TNR_tst	TNR*TPR Test
64	0,601723	0,806269	0,485151
190	0,635175	0,773308	0,491186
1024	0,627896	0,756297	0,474876
2048	0,624648	0,759753	0,474578

To higher mappers, the lowest TPR (relevant!)

ECBDL'14 Big Data Competition



ECBDL'14 Big Data Competition



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
190	0,635175	0,773308	0,491186

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}

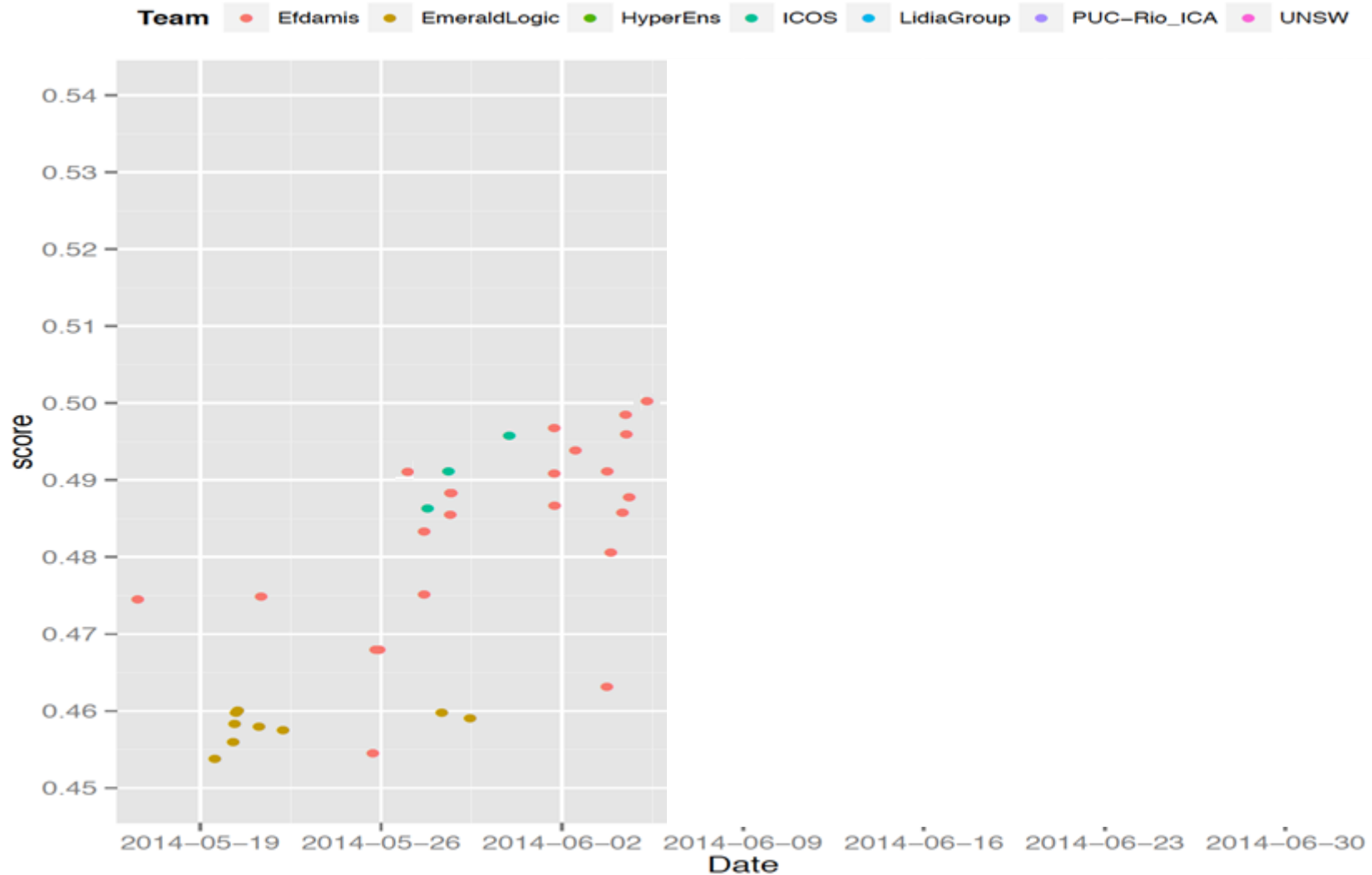
RandomForest:

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

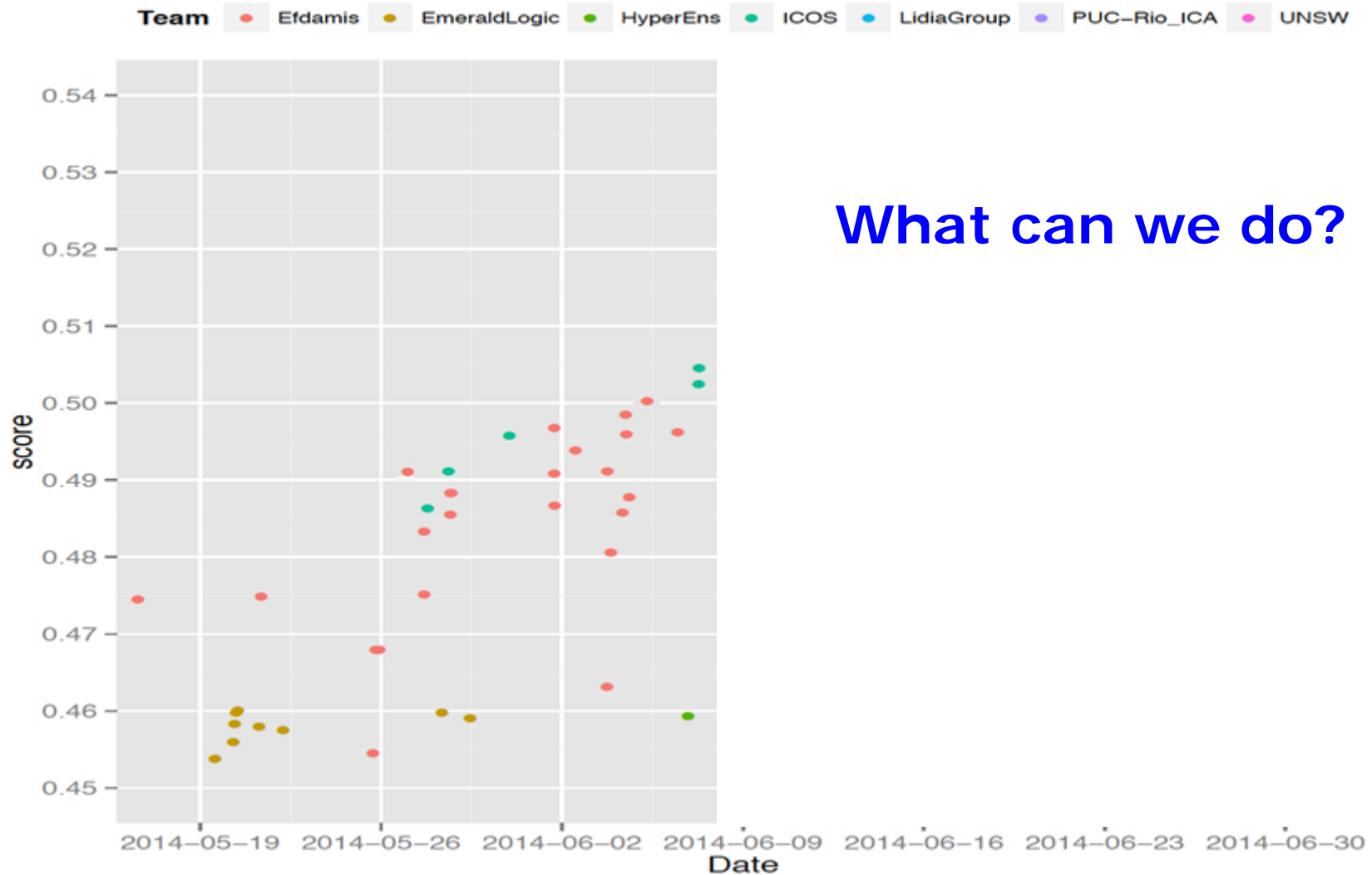
Algorithms	TPR	TNR	TNR*TPR Test
ROS+RF (RS: 100%)	0.6351	0.7733	0.491186
ROS+RF (RS: 105%)	0.6568	0.7555	0.496286
ROS+RF (RS: 110%)	0.6759	0.7337	0.495941
ROS+RF (RS: 115%)	0.7041	0.7103	0.500175
ROS+RF (RS: 130%)	0.7472	0.6609	0.493913

**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



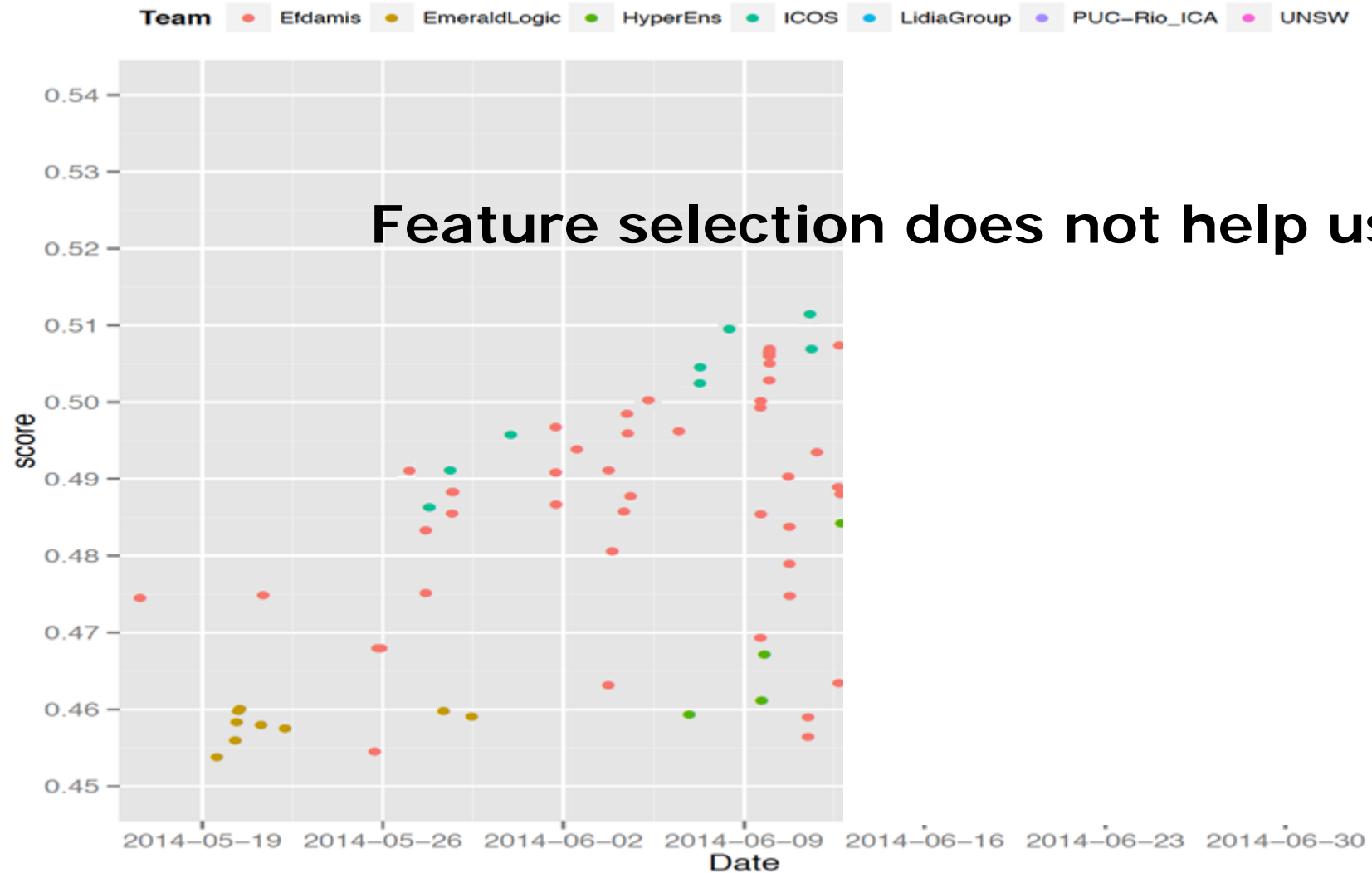
3. Detect relevant features.

1. Evolutionary Feature Selection

Classifying test set.



ECBDL'14 Big Data Competition



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

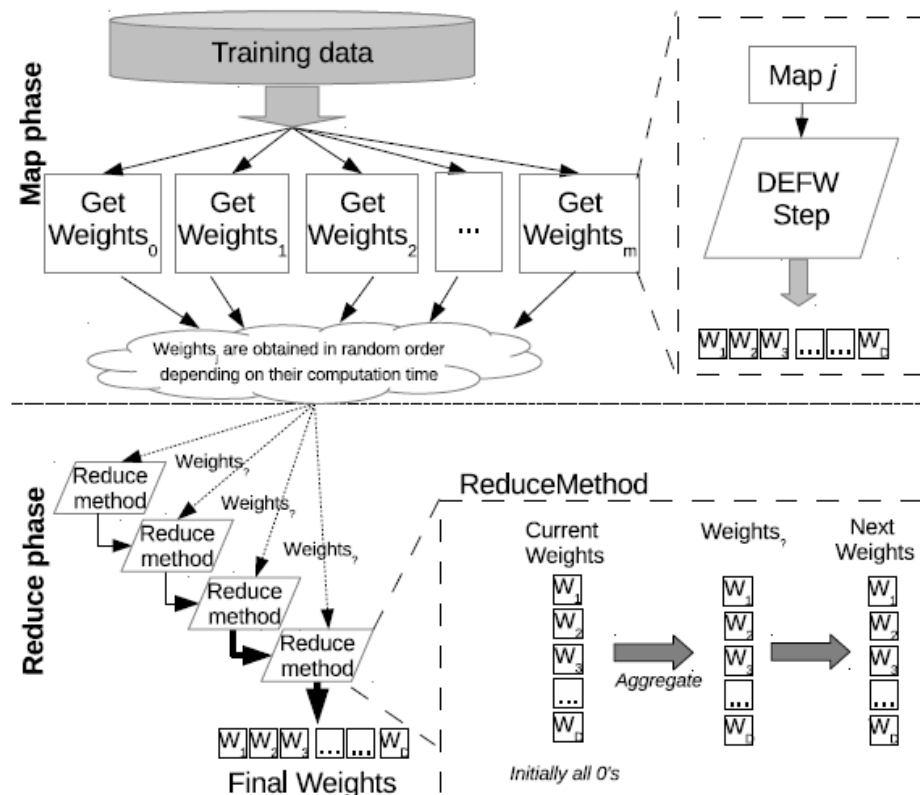
- ❑ Aggregate the feature's weights
- ❑ A feature is finally selected if it overcomes a given threshold.
- ❑ **Output:** a binary vector that represents the final selection

I. Triguero, J. Derrac, S. García, F. Herrera, Integrating a Differential Evolution Feature Weighting scheme into Prototype Generation. *Neurocomputing* 97 (2012) 332-343

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

Experimental study

Random Oversampling:

- ❑ Oversampling ratio. Analyzed values: {100 to 130}

Feature Weigthing:

- ❑ Threshold --> number of selected features.
- ❑ Set of features: {19, 63, 90, 146}
- ❑ Number of maps: 32768

RandomForest:

- ❑ Number of used features: { $\log \text{NumFeatures}$, $2 * \text{Log} + 1$ }
- ❑ Number of trees: {100}
- ❑ Number of maps: {32, 64, 128, 190, 256, 512}

ECBDL'14 Big Data Competition

We investigate: The use of Evolutionary Feature Weighting.
 It allows us to construct several subset of features (changing the threshold).

128 mappers				
Algorithms	TNR*TPR Training	TPR	TNR	TNR*TPR Test
ROS+RF (130% - Feature Weighting 19)	0.621638	0.684726	0.735272	0.503459
ROS+RF (115% - Feature Weighting 19)	0.628225	0.674569	0.750184	0.506051
ROS+RF (100% - Feature Weighting 19)	0.635029	0.629397	0.784132	0.493531
ROS+RF (130% - Feature Weighting 63)	0.634843	0.683800	0.756926	0.517586
ROS+RF (115% - Feature Weighting 63)	0.639319	0.677015	0.764589	0.517638
ROS+RF (100% - Feature Weighting 63)	0.648723	0.638567	0.794595	0.507402
64 mappers				
Algorithms	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

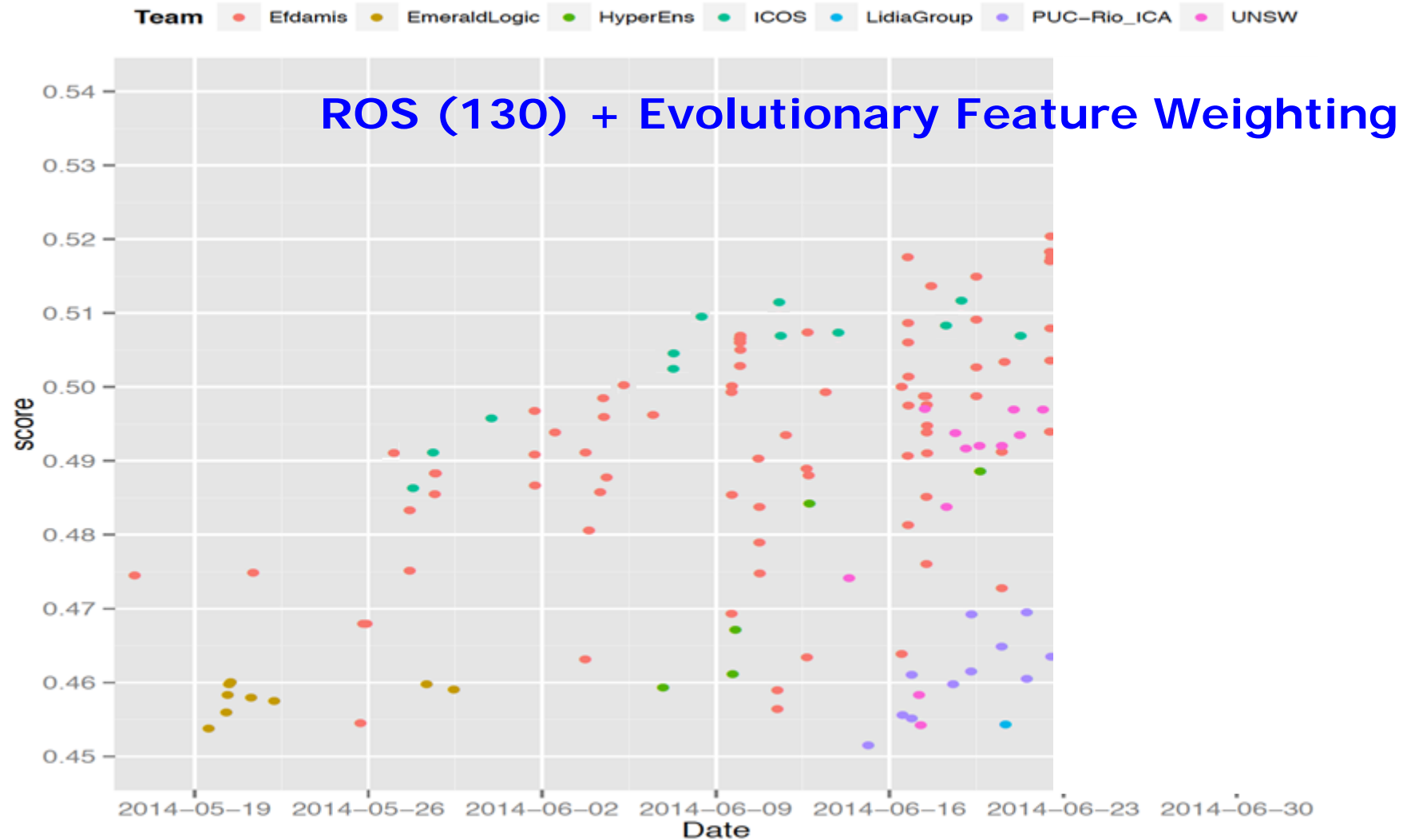
ECBDL'14 Big Data Competition

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

More features with different Maps configuration

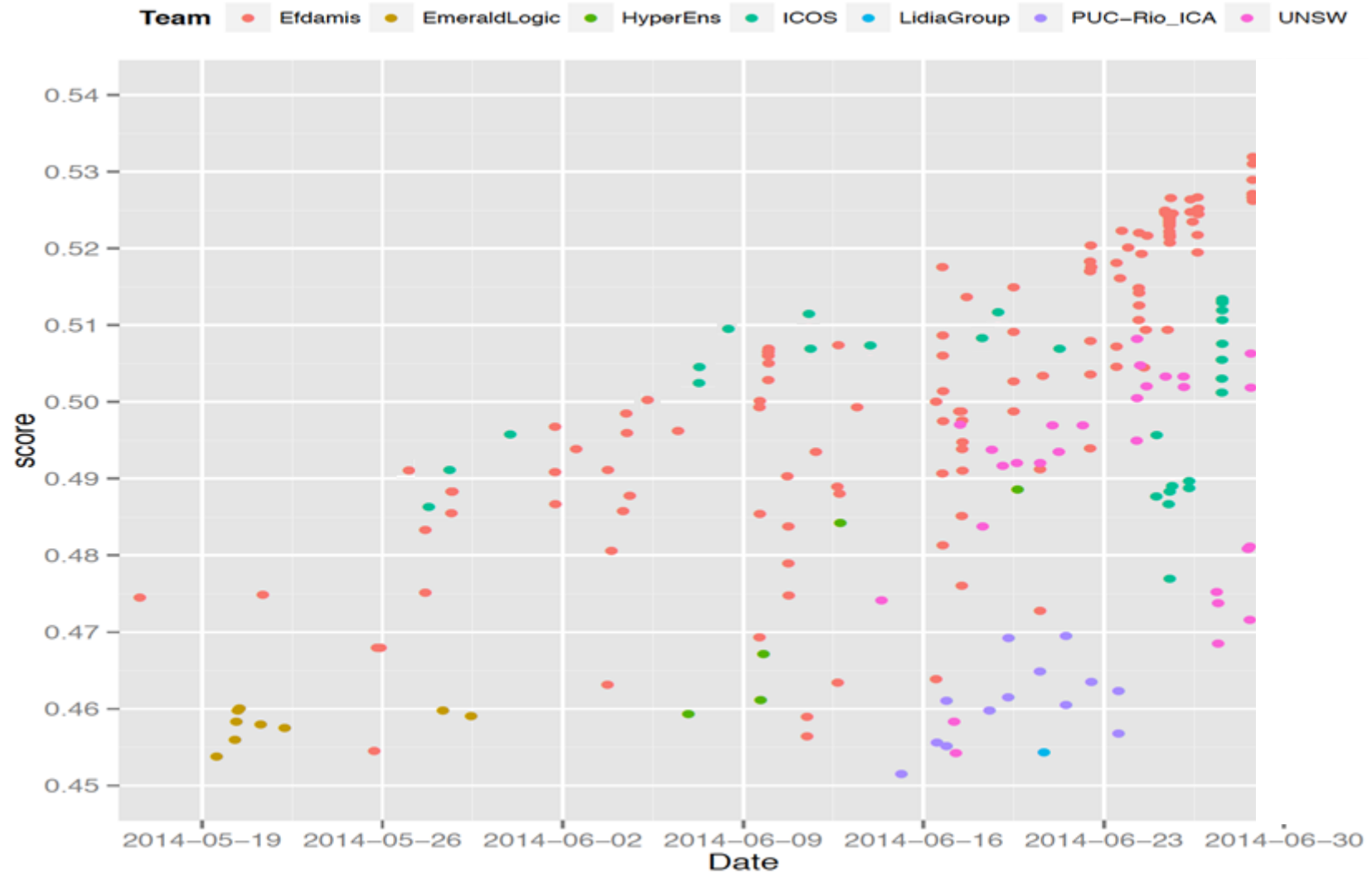
Algorithms	190 mappers			
	TNR*TPR Training	TPR	TNR	TNR*TPR Test
ROS+ RF (140%+ FW 90+25f+200t)	0.629273	0.721652	0.729740	0.526618

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

64 mappers and we got 0.53

ROS 130 – 65 replications of the minority instances
(ROS 140 – 68)

ECBDL'14 Big Data Competition



ECBDL'14 Big Data Competition

Current state:

Algorithms	64 mappers			
	TNR*TPR Training	TPR	TNR	TNR*TPR Test
ROS+ RF (140%+ FW 90+25f+200t)	0.717048	0.695109	0.763951	0.531029

Our knowledge:

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

The less number of maps, the less TPR and the high TNR (high accuracy).

ROS 130 – 65 (140 – 68) replications of the minority instances

4 days to finish the competition:

Can we take decisions for improving the model?

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

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	0.632000	0.735545	0.733808	0.464871

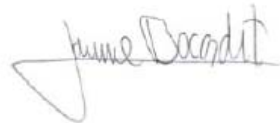
EFDAMIS team ranked first in the ECBDL'14 big data competition

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

ECBDL'14 Big Data Competition

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



ECBDL'14 Big Data Competition

Final comments

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 **ROSEFW-RF algorithm**

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

ECBDL'14 Big Data Competition

Final comments

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

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

To increase ROS and to use Evolutionary feature weighting were two good decisions for getting the first position

ECBDL'14 Big Data Competition

Final comments

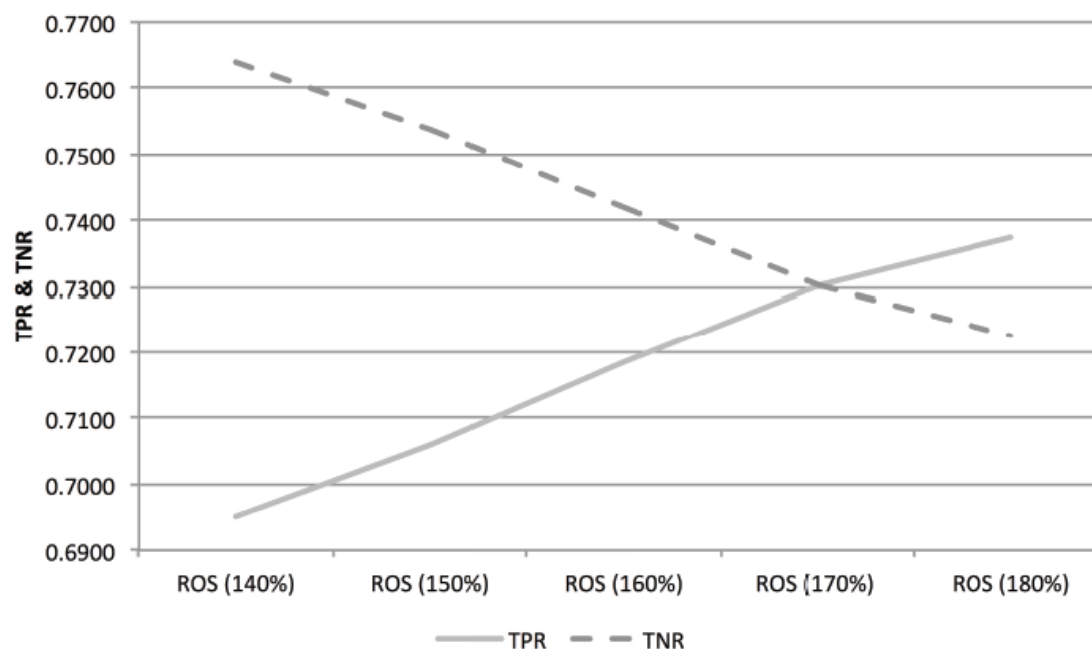


Figure 8: TPR vs. TNR varying the ROS percentage

Experiments with 64 maps

ROS 170 – 85 replications of the minority instances

Remember the initial problem. Lack of density of the minority class

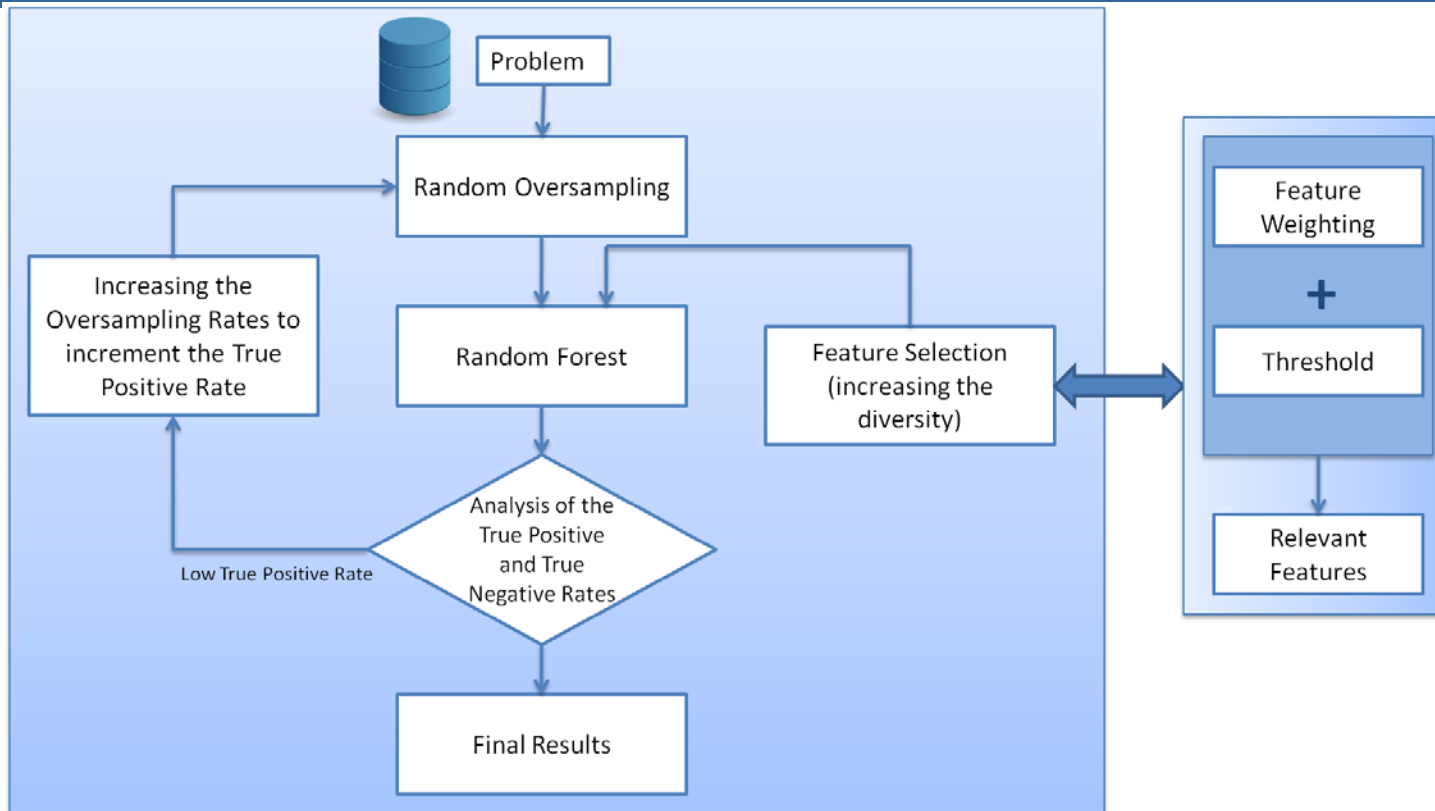
ECBDL'14 Big Data Competition

Final comments

Team Name	Learning strategy	Computational Infrastructure
Efdamis	Oversampling + EFW + Random Forest	MapReduce
ICOS	Oversampling + Ensemble of Rule sets	Batch HPC
UNSW	Ensemble of Deep Learning classifiers	Parallel HPC
HyperEns	SVM	Parallel HPC
PUC-Rio_ICA	Linear GP	GPUs
EmeraldLogic	~Linear GP	GPUs
LidiaGroup	1-layer NN	Spark

ECBDL'14 Big Data Competition

Our algorithm: **ROSEFW-RF**



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, 2015, In press.

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



Outline

IFSA – EUSFLAT 2015
Celebrating fifty years of Fuzzy Sets

- ❑ Big Data. Big Data Science
- ❑ Why Big Data? MapReduce Paradigm. Hadoop Ecosystem
- ❑ Big Data Classification: Learning algorithms
- ❑ Big Data Classification: Computational Intelligence Approches
- ❑ Big Data Classification: Imbalanced classes
- ❑ Final Comments

Final Comments



Data Mining, Machine learning and data preprocessing: Huge collection of algorithms



Big Data: A small subset of algorithms



Big Data Preprocessing:
A few methods for preprocessing in
Big Data analytics.

Final Comments



- ❑ **Fuzzy models for big data:** Robustness to the lack of data for the data fragmentation, increasing the final number of rules, and produce a high performance (accuracy).
- ❑ **The focus should be on the combination phase (reduce).**
The combination of models is the challenge for each algorithm

Final Comments



Some Challenges on Big Data Classification

❑ Computing Model

- ❑ Accuracy and Approximation
- ❑ Efficiency requirements for Algorithms

❑ Management of the uncertainty

❑ Clean Imperfect Big Data

- ❑ Noise in data distorts
- ❑ Missing values management

❑ Big Data Reduction

- ❑ To improve the efficiency in the big data analytics.
- ❑ Quality data for quality models in big data analytics

Final Comments (Our approaches)



Bird's eye view <http://sci2s.ugr.es/BigData>

Home » Thematic Web Sites » Big Data: Algorithms for Data Preprocessing, Computational Intelligence, and Imbalanced Classes

Big Data: Algorithms for Data Preprocessing, Computational Intelligence, and Imbalanced Classes



The web is organized according to the following summary:

1. Introduction to Big Data
2. Big Data Technologies: Hadoop ecosystem and Spark
3. Big Data preprocessing
4. Imbalanced Big Data classification
5. Big Data classification with fuzzy models
6. Dataset Repository
7. Literature review: surveys and overviews
8. Keynote slides
9. Links of interest



Throughout this Website, we have also included the **source code for the algorithms** associated with the former papers, as well as **new approaches that are under development**. Readers may find the implementations in the corresponding *GitHub and Spark Packages links* placed in those sections devoted to describe each framework. Both are marked with the corresponding logo:

Final Comments



Quality decisions must be based on quality big data and quality models.

Computational Intelligence may be very useful to create new high quality big data analytics models.



**A Tour on Big Data Classification.
Selected Computational
Intelligence approaches**

