Aproximación práctica a la ciencia de los datos y Big Data

Mahout

José Manuel Benítez Sánchez

Contenido

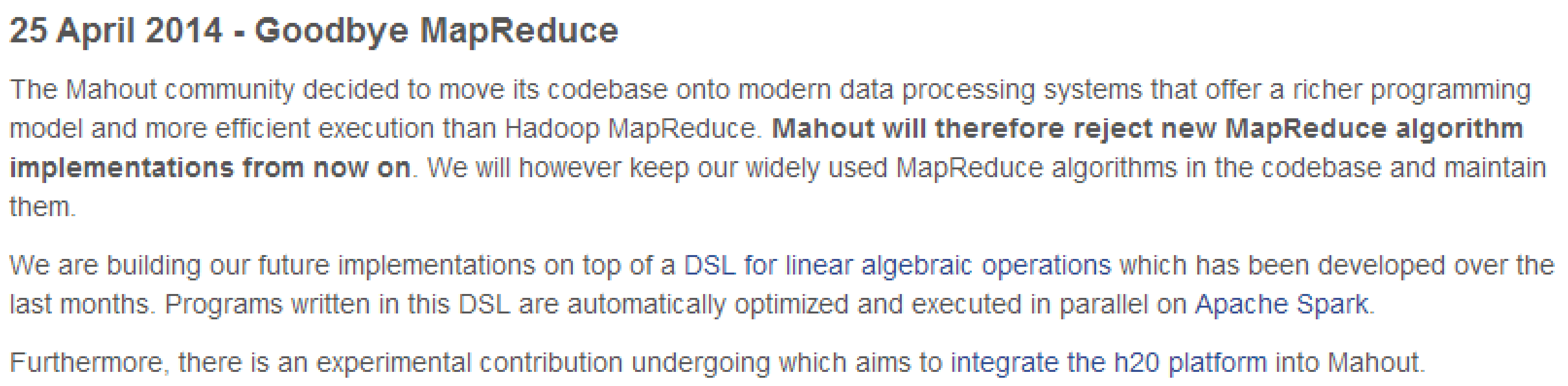
Concepto y objetivos

Apache Mahout

- Proyecto de software de la Fundación Apache Software para crear una biblioteca de aprendizaje automático (e inteligencia colectiva) **escalable** (con licencia de Software Apache)
- Está desarrollada en Java y se distribuye con licencia Apache
- Para ofrecer implementaciones escalables se apoya en *Hadoop*
- http://mahout.apache.org

Concepto

- Building a scalable machine learning library
- Dependencia original en la filosofía MapReduce de



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.

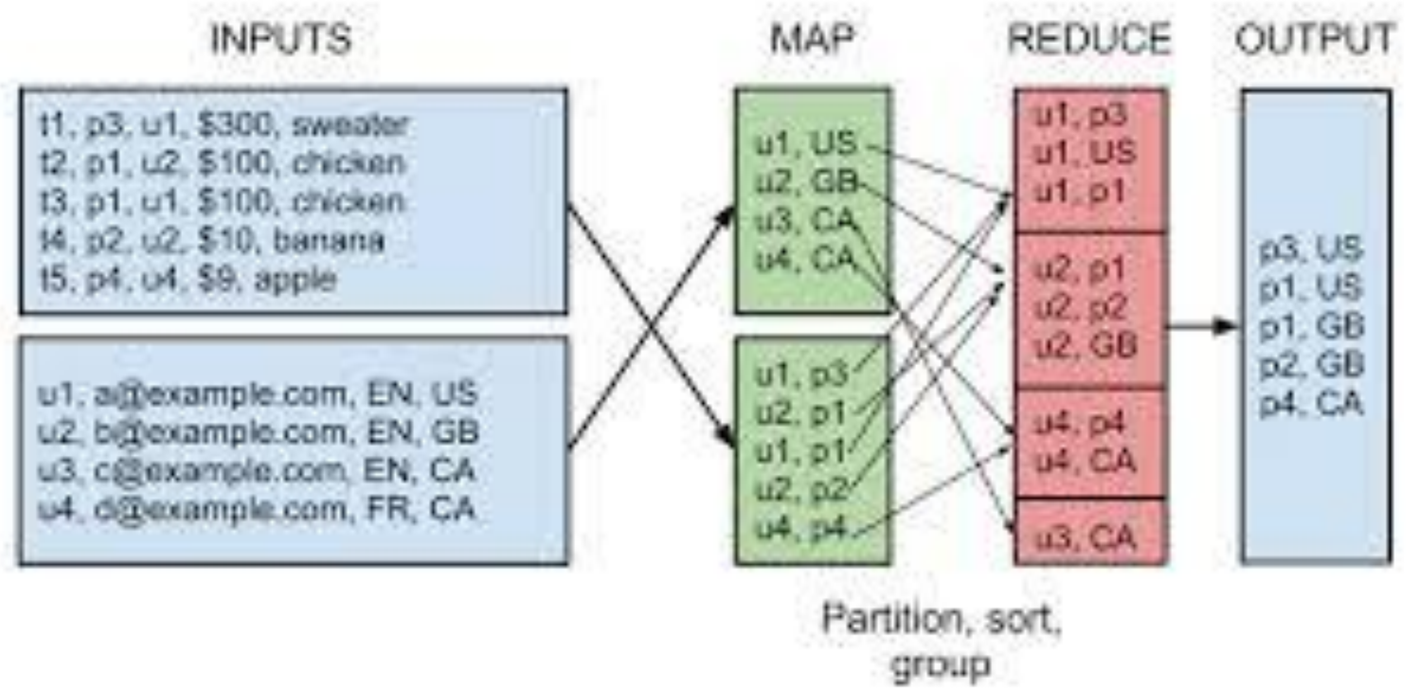
We are building our future implementations on top of a DSL for linear algebraic operations which has been developed over the last months. Programs written in this DSL are automatically optimized and executed in parallel on Apache Spark.

Furthermore, there is an experimental contribution undergoing which aims to integrate the h20 platform into Mahout.

Sentido de la “escalabilidad”

- Objetivo: ser tan rápido y eficiente como sea posible aprovechando los recursos hardware disponibles
- Depende de la naturaleza de los algoritmos:
  - Algunos simplemente no pueden escalarse
  - Otros se adecúan a la filosofía MapReduce
  - Otros necesitan modelos de programación distribuida diferentes
  - Otros son ya suficientemente rápidos

MapReduce



INPUTS
MAP
REDUCE
OUTPUT
t1, p3, u1, $300, sweater
t2, p1, u2, $100, chicken
t3, p1, u1, $100, chicken
t4, p2, u2, $10, banana
t5, p4, u4, $9, apple
u1, a@example.com, EN, US
u2, b@example.com, EN, GB
u3, c@example.com, EN, CA
u4, d@example.com, FR, CA
u1, US
u2, GB
u3, CA
u4, CA
u1, p3
u2, p1
u1, p1
u2, p2
u4, p4
u1, p3
u1, US
u1, p1
u2, p1
u2, p2
u2, GB
u4, p4
u4, CA
u3, CA
p3, US
p1, US
p1, GB
p2, GB
p4, CA
Partition, sort, group

Funcionalidad de Hadoop

- Gestionar almacenamiento (HDFS) de entradas, pares intermedios y salidas
- Particionamiento de datos
- Transferencias
- Planificación y monitorización de procesos
- Entorno de ejecución distribuida con bajo conocimiento técnico

Entornos de ejecución

- Hadoop (y Mahout sobre él) se pueden ejecutar en:
  - Modo individual (un sólo equipo; máquina virtual)
  - Cluster de ordenadores
  - Plataformas de Cloud Computing

Algunos usuarios de Mahout

foursquare

Booz | Allen | Hamilton

Adobe

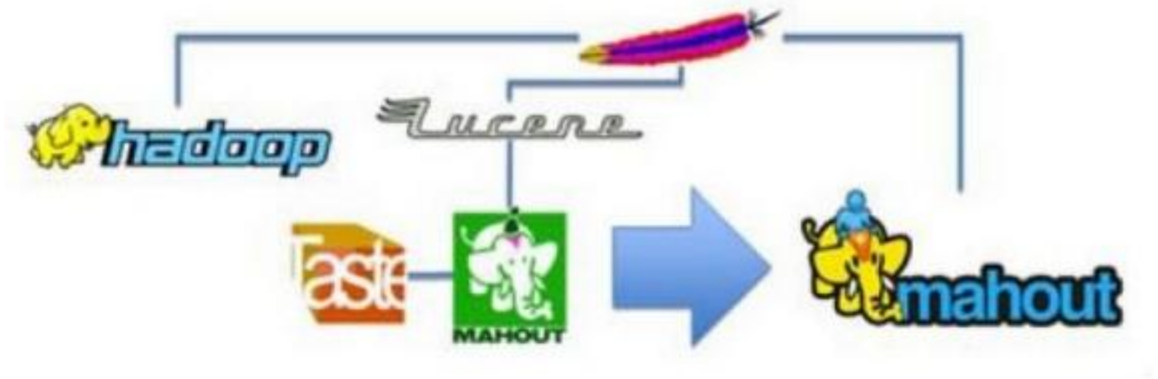
BUZZLOGIC
WELCOME TO THE CONVERSATION

Shop It To Me

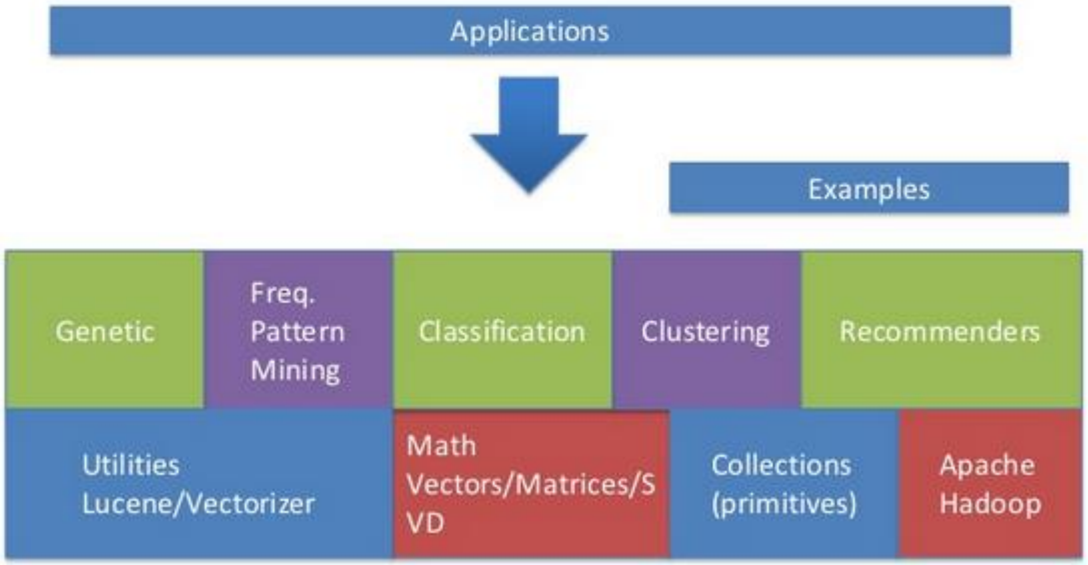
twitter

YAHOO!

Mahout dentro de Apache Foundation



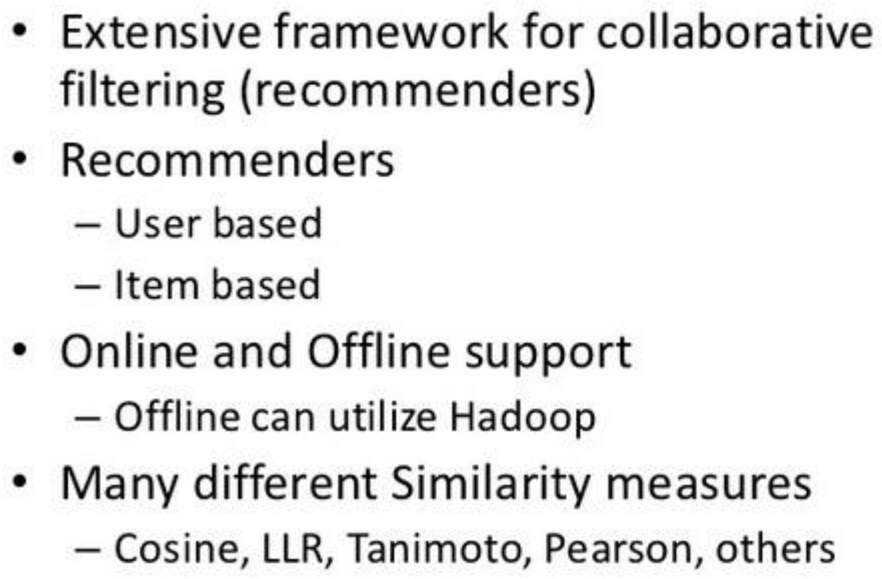
Arquitectura



Funcionalidad incluida

- Clustering
- Clasificación
- Análisis de patrones frecuentes
- Sistemas de recomendación (Filtro colaborativo)

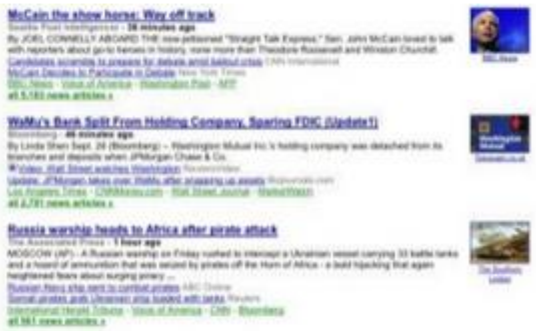
Filtrado colaborativo



- Extensive framework for collaborative filtering (recommenders)
- Recommenders
  - User based
  - Item based
- Online and Offline support
  - Offline can utilize Hadoop
- Many different Similarity measures
  - Cosine, LLR, Tanimoto, Pearson, others

Clustering

- Document level
  - Group documents based on a notion of similarity
  - K-Means, Fuzzy K-Means, Dirichlet, Canopy, Mean-Shift, EigenCuts (Spectral)
  - All Map/Reduce
  - Distance Measures
    - Manhattan, Euclidean, other



McCain the show horse: Way off track
Seattle Post Intelligencer - 38 minutes ago
By JOEL CONNELLY ABOARD THE now-jettisoned "Straight Talk Express," Sen. John McCain loved to talk with reporters about go-to heroes in history, none more than Theodore Roosevelt and Winston Churchill.
Candidates scramble to prepare for debate amid bailout crisis CNN International
McCain Decides to Participate in Debate New York Times
BBC News - Voice of America - Washington Post - AFP
all 5,183 news articles »

WaMu's Bank Split From Holding Company, Sparing FDIC (Update1)
Bloomberg - 46 minutes ago
By Linda Shen Sept. 26 (Bloomberg) -- Washington Mutual Inc.'s holding company was detached from its branches and deposits when JPMorgan Chase & Co.
Video: Wall Street watches Washington Reuters Video
Update: JPMorgan takes over WaMu after snapping up assets Bizjournals.com
Los Angeles Times - CNNMoney.com - Wall Street Journal - MarketWatch
all 2,781 news articles »

Russia warship heads to Africa after pirate attack
The Associated Press - 1 hour ago
MOSCOW (AP) - A Russian warship on Friday rushed to intercept a Ukrainian vessel carrying 33 battle tanks and a hoard of ammunition that was seized by pirates off the Horn of Africa - a bold hijacking that again heightened fears about surging piracy ...
Russian Navy ship sent to combat pirates ABC Online
Somali pirates grab Ukrainian ship loaded with tanks Reuters
International Herald Tribune - Voice of America - CNN - Bloomberg
all 561 news articles »

- Topic Modeling
  - Cluster words across documents to identify topics
  - Latent Dirichlet Allocation (M/R)

Clasificación

- Place new items into predefined categories:
  - Sports, politics, entertainment
  - Recommenders
- Implementations
  - Naïve Bayes (M/R)
  - Compl. Naïve Bayes (M/R)
  - Decision Forests (M/R)
  - Linear Regression (Seq. but Fast!)

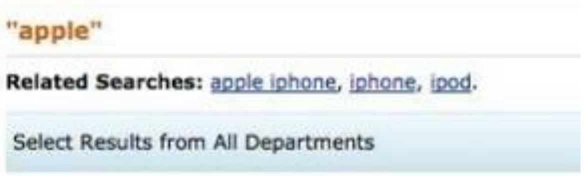
Shop It To Me

•See Chapter 17 of Mahout in Action for Shop It To Me use case:

•http://awe.sm/5FyNe

Minería de patrones frecuentes

- Identify frequently co-occurrent items
- Useful for:
  - Query Recommendations
    - Apple -> iPhone, orange, OS X
  - Related product placement
    - Basket Analysis
- Map/Reduce


"apple"

Related Searches: apple iphone, iphone, ipod.

Select Results from All Departments

http://www.amazon.com

Fuentes de información

- Integración con Lucene:
  - bin/mahout lucevector
- Vectorizador de documentos
  - bin/mahout seqdirectory …
  - bin/mahout seq2sparse …
- Programmatically
  - Utils module in Mahout
- Database
- Filesystem:
  - Regular
  - HDFS

Instalación

https://mahout.apache.org/users/basics/quickstart.html

http://apache.rediris.es/mahout/

mahout-distribution-0.9-src.zip

mahout-distribution-0.9.zip

Dependencias

- Java (Oracle JDK)
- Maven
- Hadoop

Estructura de la distribución

- bin
- buildtools
- core
- distribution
- examples
- integration
- math
- math-scala
- src
- target

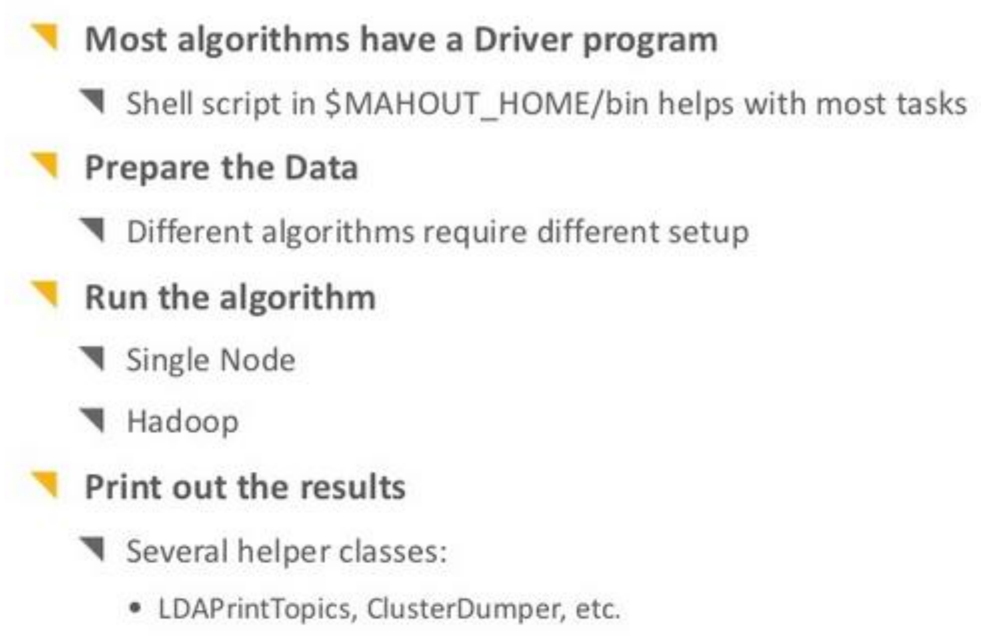
Compilación, configuración, desarrollo

- Compilación: mvn –DskipTests clean install
- Variables de entorno:
  - JAVA_HOME
  - HADOOP_HOME
  - MAVEN_HOME
  - PATH
- Se recomienda el desarrollo con un IDE:
  - Eclipse
  - NetBeans
  - Emacs, …

Modos de uso

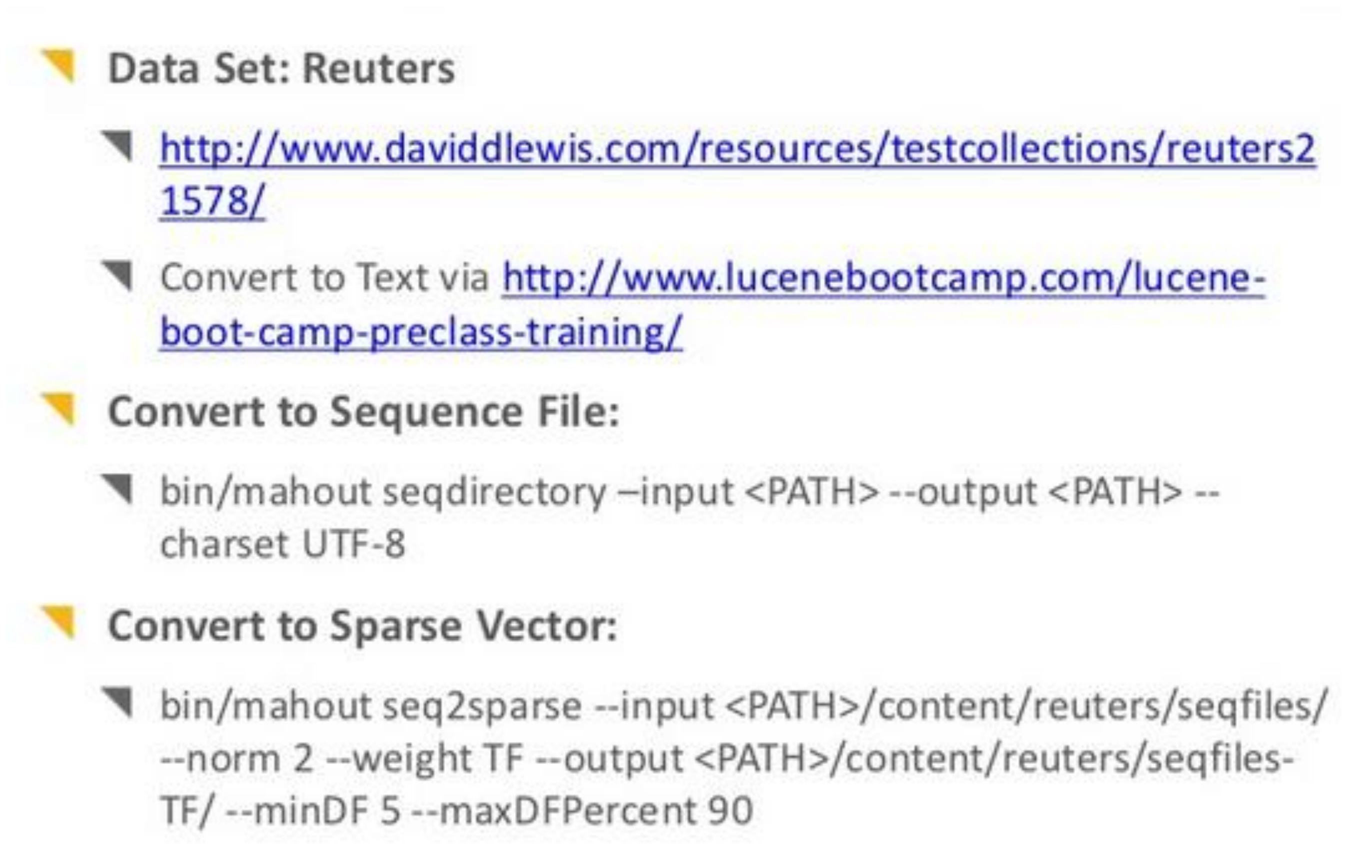
- Mahout es una biblioteca, no una aplicación con GUI
- Uso con mandatos desde línea de órdenes
- Uso con programas que invocan a la biblioteca:
  - Java
  - Scala

Uso en línea de órdenes



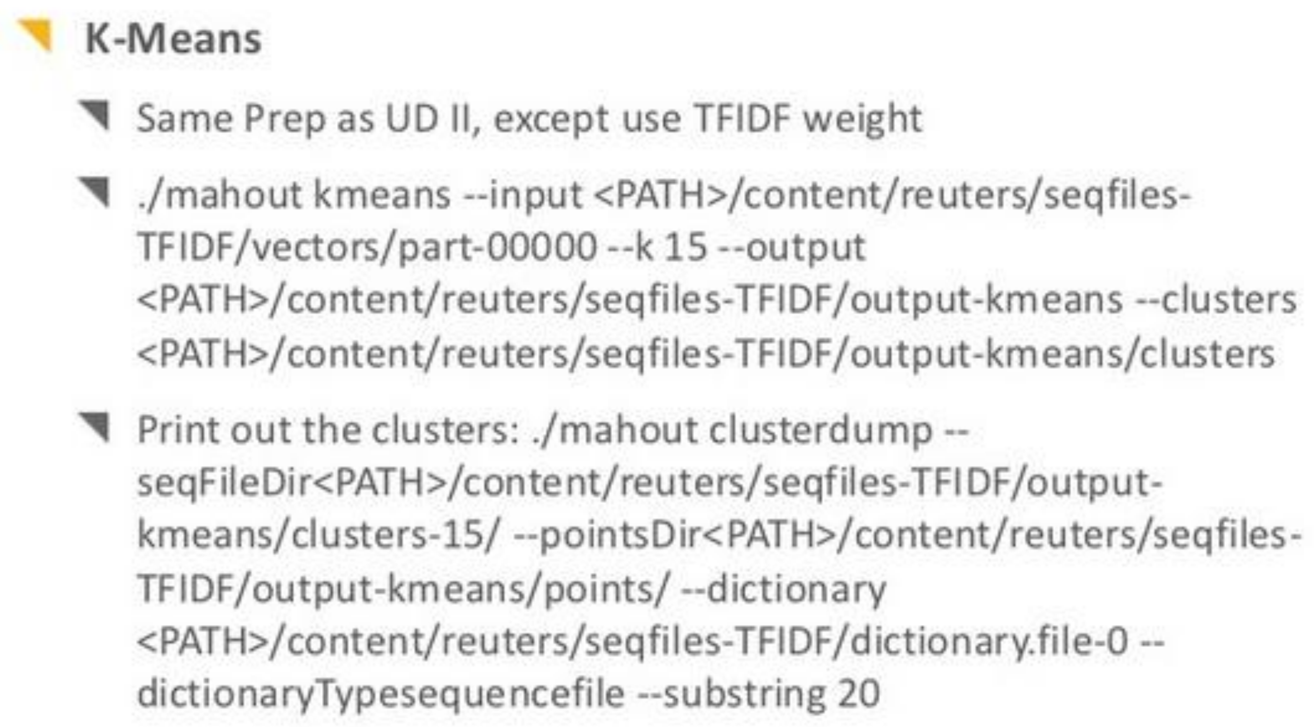
- **Most algorithms have a Driver program**
  - Shell script in $MAHOUT_HOME/bin helps with most tasks
- **Prepare the Data**
  - Different algorithms require different setup
- **Run the algorithm**
  - Single Node
  - Hadoop
- **Print out the results**
  - Several helper classes:
    - LDAPrintTopics, ClusterDumper, etc.

Preparación de datos



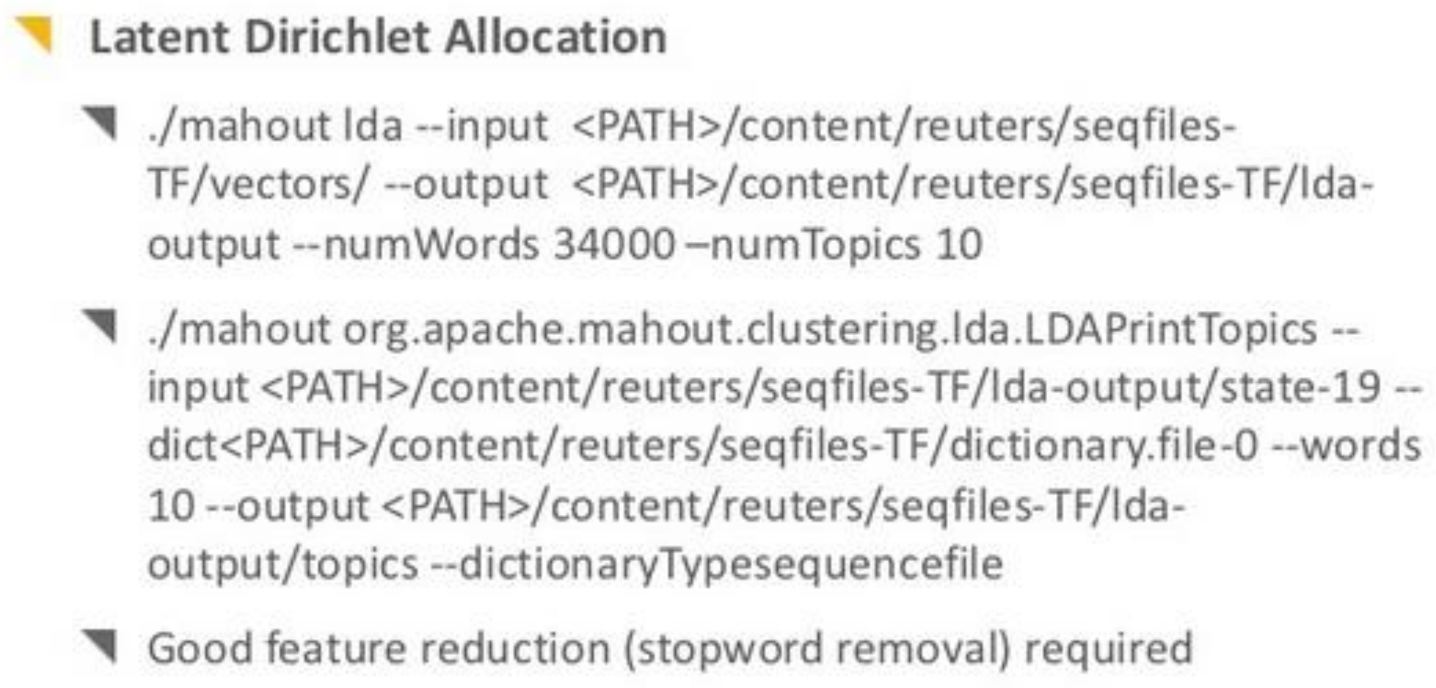
- **Data Set: Reuters**
  - http://www.daviddlewis.com/resources/testcollections/reuters21578/
  - Convert to Text via http://www.lucenebootcamp.com/lucene-boot-camp-preclass-training/
- **Convert to Sequence File:**
  - bin/mahout seqdirectory –input <PATH> --output <PATH> --charset UTF-8
- **Convert to Sparse Vector:**
  - bin/mahout seq2sparse --input <PATH>/content/reuters/seqfiles/ --norm 2 --weight TF --output <PATH>/content/reuters/seqfiles-TF/ --minDF 5 --maxDFPercent 90

Clustering: K-Means



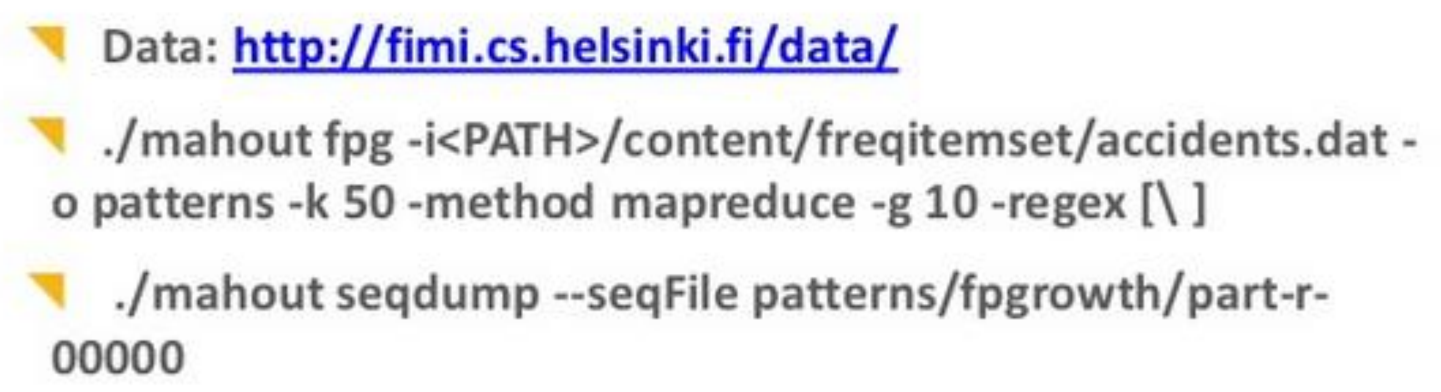
- **K-Means**
  - Same Prep as UD II, except use TFIDF weight
  - ./mahout kmeans --input <PATH>/content/reuters/seqfiles-TFIDF/vectors/part-00000 --k 15 --output <PATH>/content/reuters/seqfiles-TFIDF/output-kmeans --clusters <PATH>/content/reuters/seqfiles-TFIDF/output-kmeans/clusters
  - Print out the clusters: ./mahout clusterdump --seqFileDir<PATH>/content/reuters/seqfiles-TFIDF/output-kmeans/clusters-15/ --pointsDir<PATH>/content/reuters/seqfiles-TFIDF/output-kmeans/points/ --dictionary <PATH>/content/reuters/seqfiles-TFIDF/dictionary.file-0 --dictionaryTypesequencefile --substring 20

Clasificación: LDA



- **Latent Dirichlet Allocation**
  - ./mahout lda --input <PATH>/content/reuters/seqfiles-TF/vectors/ --output <PATH>/content/reuters/seqfiles-TF/lda-output --numWords 34000 –numTopics 10
  - ./mahout org.apache.mahout.clustering.lda.LDAPrintTopics --input <PATH>/content/reuters/seqfiles-TF/lda-output/state-19 --dict<PATH>/content/reuters/seqfiles-TF/dictionary.file-0 --words 10 --output <PATH>/content/reuters/seqfiles-TF/lda-output/topics --dictionaryTypesequencefile
  - Good feature reduction (stopword removal) required

Minería de patrones frecuentes



- Data: http://fimi.cs.helsinki.fi/data/
- ./mahout fpg -i<PATH>/content/freqitemset/accidents.dat -o patterns -k 50 -method mapreduce -g 10 -regex [\ ]
- ./mahout seqdump --seqFile patterns/fpgrowth/part-r-00000

Preparación de datos

- Cada algoritmo tiene requisitos particulares con respecto al formato de los datos que acepta
- Sequence File
- Vectorizados: representación en vectores
  - DenseVector
  - RandomAccessSparseVector
  - SequentialAccessSparseVector

Sequence File

- Fichero binario que codifica la información en pares <clave, valor>. Formato específico de Hadoop
- Clave debe implementar `WritableComparable`
- Valor debe implementar `Writable`
- El fichero incluye una cabecera con metadatos:
  - Versión
  - Nombre de la clave
  - Nombre del valor
  - Compresión

Clustering

Elementos básicos para el clustering

- Un algoritmo
- Medida de similitud o distancia
- Condición de parada

Medidas de distancia

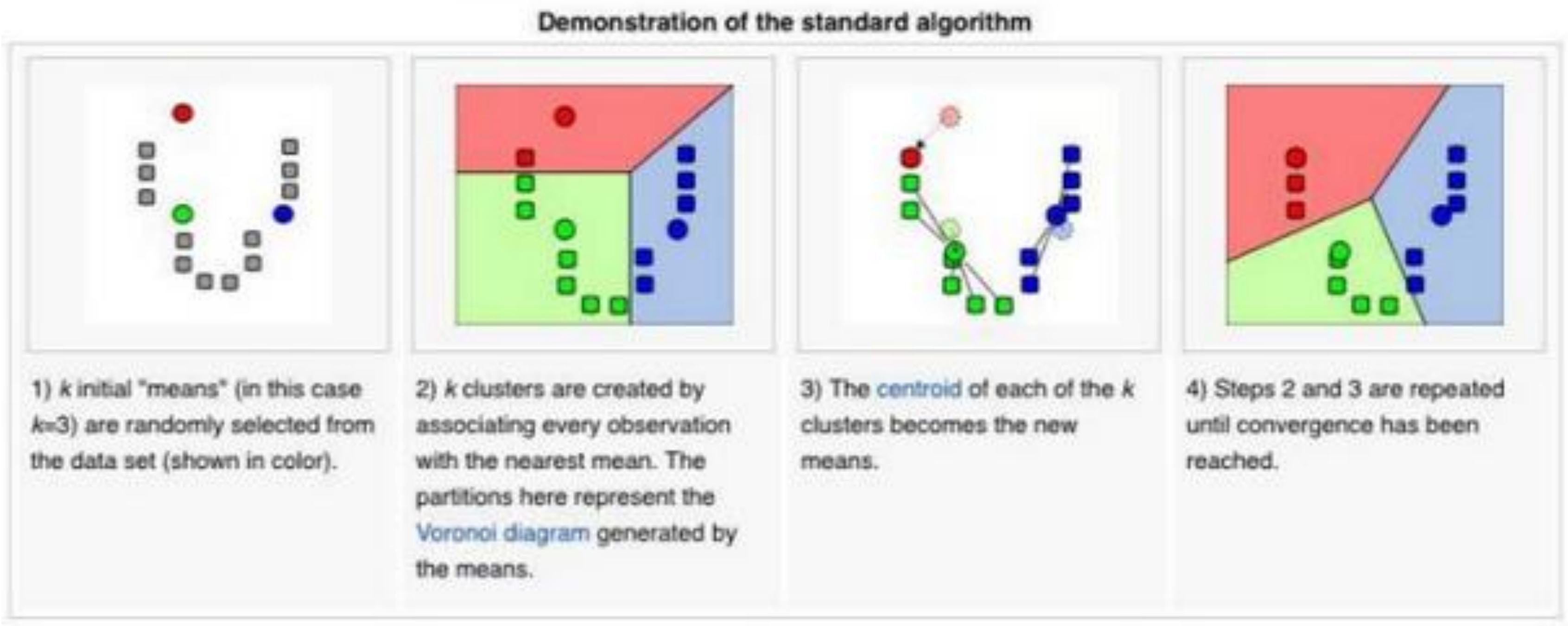
- Euclidean distance measure
- Squared Euclidean distance measure
- Manhattan distance measure
- Cosine distance measure
- Tanimoto distance measure
- Weighted distance measure

Algoritmos de clustering en Mahout

- K-means
- Canopy
- Fuzzy k-Means
- Streaming k-Means
- Spectral clustering (Model-based)

K-Means

- Clustering Algorithm
  - Nicely parallelizable!



Demonstration of the standard algorithm

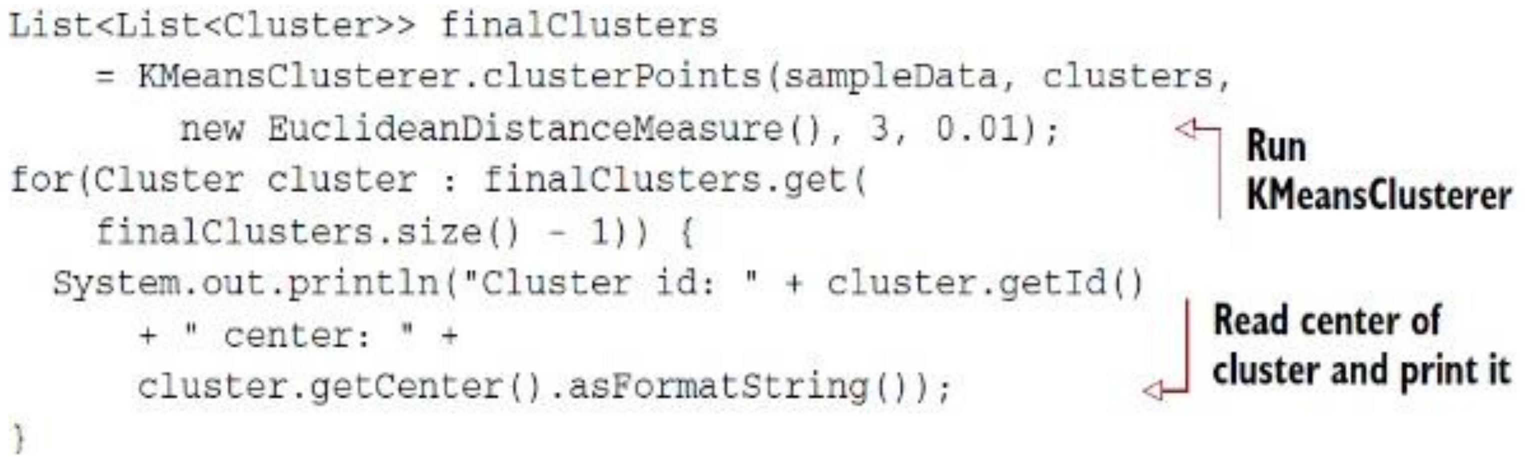
1) *k* initial "means" (in this case *k*=3) are randomly selected from the data set (shown in color).

2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

3) The centroid of each of the *k* clusters becomes the new means.

4) Steps 2 and 3 are repeated until convergence has been reached.

Desde código fuente

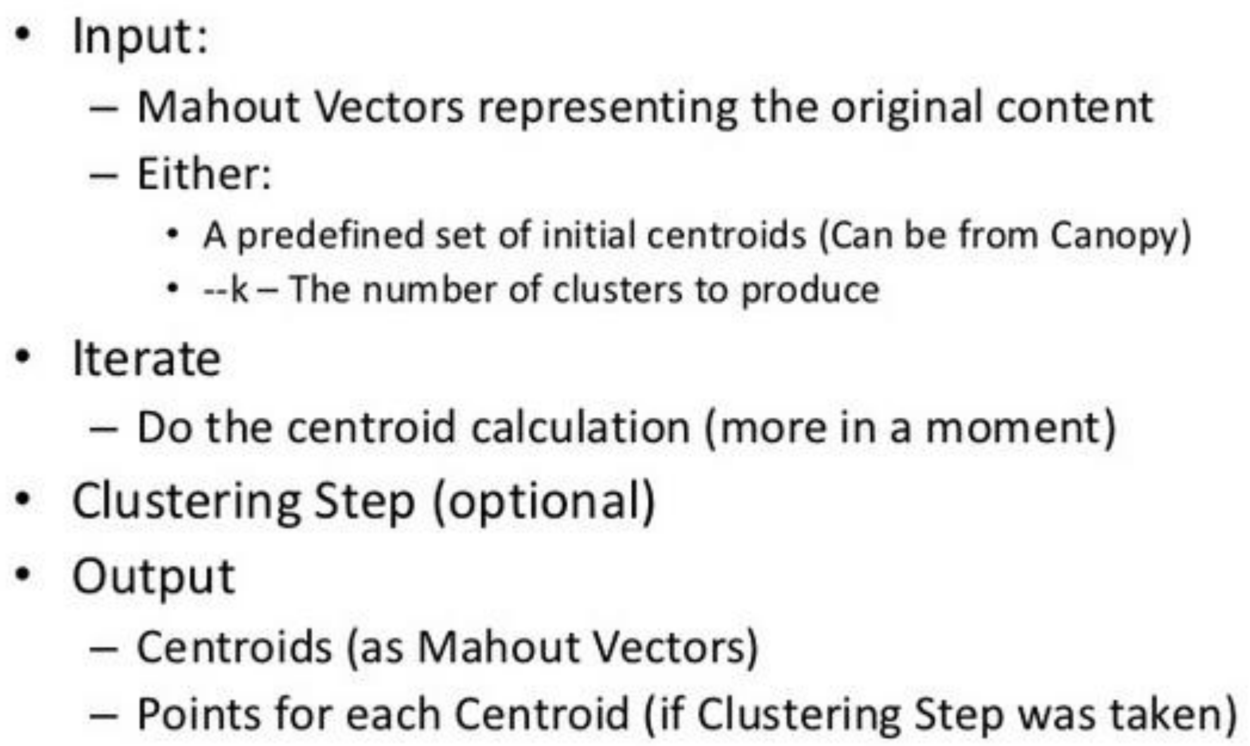


```
List<List<Cluster>> finalClusters
    = KMeansClusterer.clusterPoints(sampleData, clusters,
        new EuclideanDistanceMeasure(), 3, 0.01);
for(Cluster cluster : finalClusters.get(
    finalClusters.size() - 1)) {
  System.out.println("Cluster id: " + cluster.getId()
      + " center: " +
      cluster.getCenter().asFormatString());
}
```

Run KMeansClusterer

Read center of cluster and print it

K-Means en MapReduce



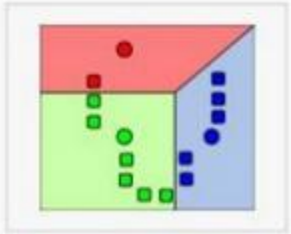
- Input:
  - Mahout Vectors representing the original content
  - Either:
    - A predefined set of initial centroids (Can be from Canopy)
    - --k – The number of clusters to produce
- Iterate
  - Do the centroid calculation (more in a moment)
- Clustering Step (optional)
- Output
  - Centroids (as Mahout Vectors)
  - Points for each Centroid (if Clustering Step was taken)

MapReduce Iteration

- Each Iteration calculates the Centroids using:
  - KMeansMapper
  - KMeansCombiner
  - KMeansReducer
- Clustering Step
  - Calculate the points for each Centroid using:
  - KMeansClusterMapper

KMeansMapper

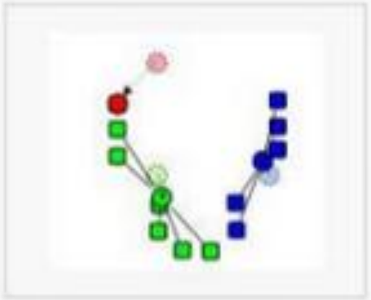
- During Setup:
  - Load the initial Centroids (or the Centroids from the last iteration)
- Map Phase
  - For each input
    - Calculate it's distance from each Centroid and output the closest one
- Distance Measures are pluggable
  - Manhattan, Euclidean, Squared Euclidean, Cosine, others

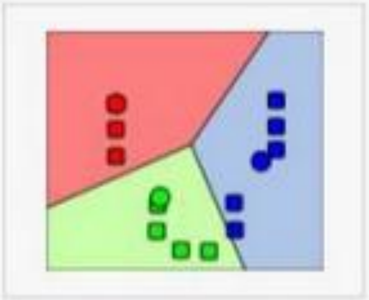


2) *k* clusters are created by associating every observation with the nearest mean. The partitions here represent the Voronoi diagram generated by the means.

KMeansReducer

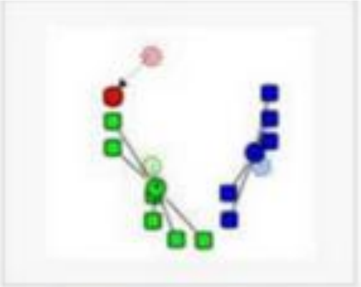
- Setup:
  - Load up clusters
  - Convergence information
  - Partial sums from KMeansCombiner (more in a moment)
- Reduce Phase
  - Sum all the vectors in the cluster to produce a new Centroid
  - Check for Convergence
- Output cluster


3) The centroid of each of the *k* clusters becomes the new means.


4) Steps 2 and 3 are repeated until convergence has been reached.

KMeansCombiner

- Just like KMeansReducer, but only produces partial sum of the cluster based on the data local to the Mapper



3) The centroid of each of the *k* clusters becomes the new means.

KMeansClusterMapper

- Some applications only care about what the Centroids are, so this step is optional

- Setup:
  - Load up the clusters and the DistanceMeasure used
- Map Phase
  - Calculate which Cluster the point belongs to
  - Output <ClusterId, Vector>

Pasos para procesar datos con clustering

1. Preprocesar los datos
2. Crear vectores a partir de los datos
3. Almacenar los vectores en un SequenceFile

Ejemplo sencillo (1/4)

```
public static final double[][] points = { {1, 1}, {2, 1}, {1, 2},
                                          {2, 2}, {3, 3}, {8, 8},
                                          {9, 8}, {8, 9}, {9, 9}};

public static void writePointsToFile(List<Vector> points,
                                     String fileName,
                                     FileSystem fs,
                                     Configuration conf) throws IOException {
  Path path = new Path(fileName);
  SequenceFile.Writer writer = new SequenceFile.Writer(fs, conf,
      path, LongWritable.class, VectorWritable.class);
  long recNum = 0;
  VectorWritable vec = new VectorWritable();
  for (Vector point : points) {
    vec.set(point);
    writer.append(new LongWritable(recNum++), vec);
  }
  writer.close();
}
```

Ejemplo sencillo (2/4)

```
public static List<Vector> getPoints(double[][] raw) {
  List<Vector> points = new ArrayList<Vector>();
  for (int i = 0; i < raw.length; i++) {
    double[] fr = raw[i];
    Vector vec = new RandomAccessSparseVector(fr.length);
    vec.assign(fr);
    points.add(vec);
  }
  return points;
}

public static void main(String args[]) throws Exception {

  int k = 2;

  List<Vector> vectors = getPoints(points);

  File testData = new File("testdata");
  if (!testData.exists()) {
    testData.mkdir();
  }
  testData = new File("testdata/points");
  if (!testData.exists()) {
    testData.mkdir();
  }
```

Specify number of clusters to be formed

Create input directories for data

Ejemplo sencillo (3/4)

```
Configuration conf = new Configuration();
FileSystem fs = FileSystem.get(conf);
writePointsToFile(vectors,
    "testdata/points/file1", fs, conf);

Path path = new Path("testdata/clusters/part-00000");
SequenceFile.Writer writer
    = new SequenceFile.Writer(
        fs, conf,       path, Text.class, Cluster.class);

for (int i = 0; i < k; i++) {
  Vector vec = vectors.get(i);
  Cluster cluster = new Cluster(
      vec, i, new EuclideanDistanceMeasure());
  writer.append(new Text(cluster.getIdentifier()), cluster);
}
writer.close();
```

$\longleftarrow$ **Write initial centers**

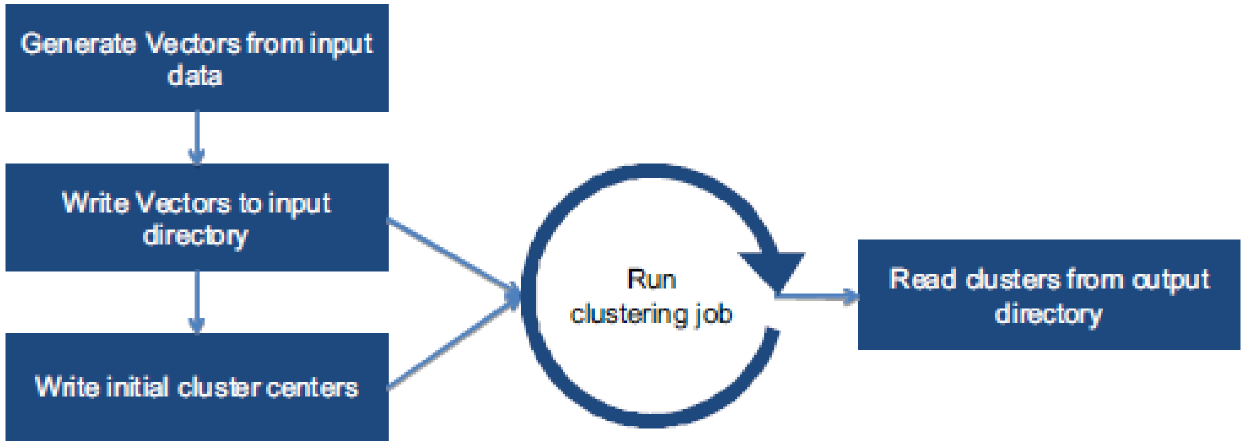
Ejemplo sencillo (4/4)

```
  KMeansDriver.run(conf, new Path("testdata/points"),
    new Path("testdata/clusters"),
    new Path("output"), new EuclideanDistanceMeasure(),
        0.001, 10, true, false);

  SequenceFile.Reader reader
      = new SequenceFile.Reader(fs,
          new Path("output/" + Cluster.CLUSTERED_POINTS_DIR
              + "/part-m-00000"), conf);

  IntWritable key = new IntWritable();
  WeightedVectorWritable value = new WeightedVectorWritable();
  while (reader.next(key, value)) {
    System.out.println(
        value.toString() + " belongs to cluster "
            + key.toString());
  }
  reader.close();
}
```

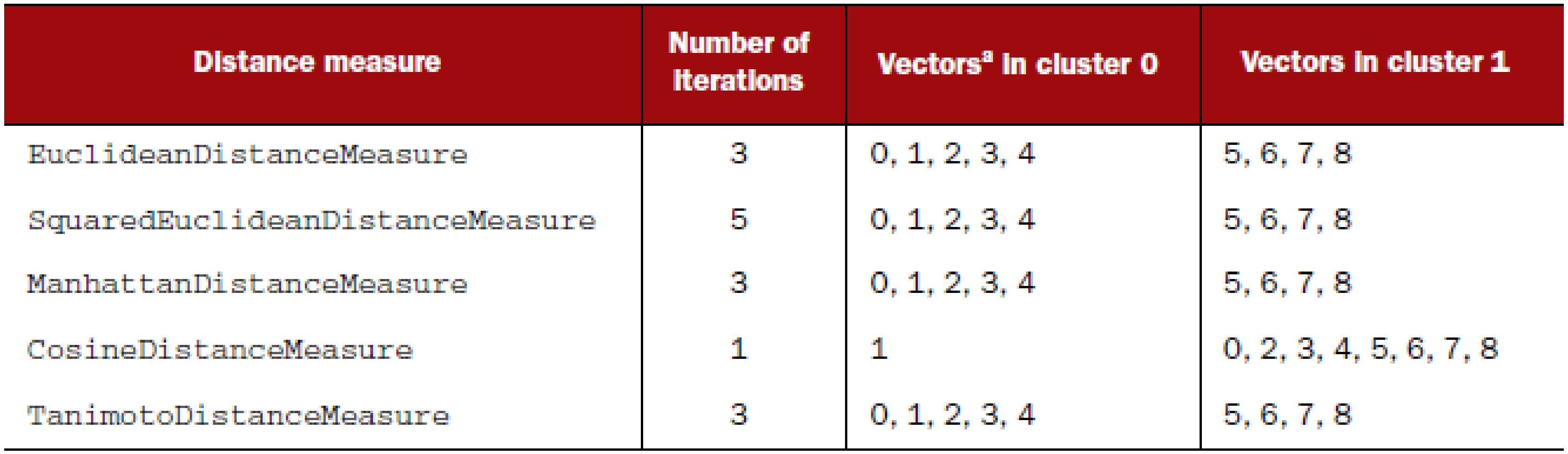
Run k-means algorithm

Read output, print vector, cluster ID

Ejemplo sencillo: diagrama de flujo


Generate Vectors from input data
Write Vectors to input directory
Write initial cluster centers
Run clustering job
Read clusters from output directory

Efecto de las métricas



Distance measure	Number of iterations	Vectors[a] in cluster 0	Vectors in cluster 1
`EuclideanDistanceMeasure`	3	0, 1, 2, 3, 4	5, 6, 7, 8
`SquaredEuclideanDistanceMeasure`	5	0, 1, 2, 3, 4	5, 6, 7, 8
`ManhattanDistanceMeasure`	3	0, 1, 2, 3, 4	5, 6, 7, 8
`CosineDistanceMeasure`	1	1	0, 2, 3, 4, 5, 6, 7, 8
`TanimotoDistanceMeasure`	3	0, 1, 2, 3, 4	5, 6, 7, 8

K-means

- El algoritmo se ejecuta usado:
  - `KmeansClusterer` (in-memory)
  - `KMeansDriver`: MapReduce

Clustering in-memory (1/2)

```java
private static void generateSamples(List<Vector> vectors, int num,
  double mx, double my, double sd) {
    for (int i = 0; i < num; i++) {
      vectors.add(new DenseVector(
        new double[] {
          UncommonDistributions.rNorm(mx, sd),
          UncommonDistributions.rNorm(my, sd)
        }
      ));
    }
  }
public static void main(String[] args) {
  List<Vector> sampleData = new ArrayList<Vector>();

  generateSamples(sampleData, 400, 1, 1, 3);
  generateSamples(sampleData, 300, 1, 0, 0.5);
  generateSamples(sampleData, 300, 0, 2, 0.1);

  int k = 3;
```

Generate three sets of points

Clustering in-memory (2/2)

```
List<Vector> randomPoints = RandomPointsUtil.chooseRandomPoints(
  sampleData, k);
   List<Cluster> clusters = new ArrayList<Cluster>();

   int clusterId = 0;
   for (Vector v : randomPoints) {
     clusters.add(new Cluster(v, clusterId++,
        new EuclideanDistanceMeasure()));
   }

   List<List<Cluster>> finalClusters
       = KMeansClusterer.clusterPoints(sampleData, clusters,
           new EuclideanDistanceMeasure(), 3, 0.01);
   for(Cluster cluster : finalClusters.get(
       finalClusters.size() - 1)) {
     System.out.println("Cluster id: " + cluster.getId()
       + " center: " +
       cluster.getCenter().asFormatString());
   }
  }
}
```

Run KMeansClusterer

Read center of cluster and print it

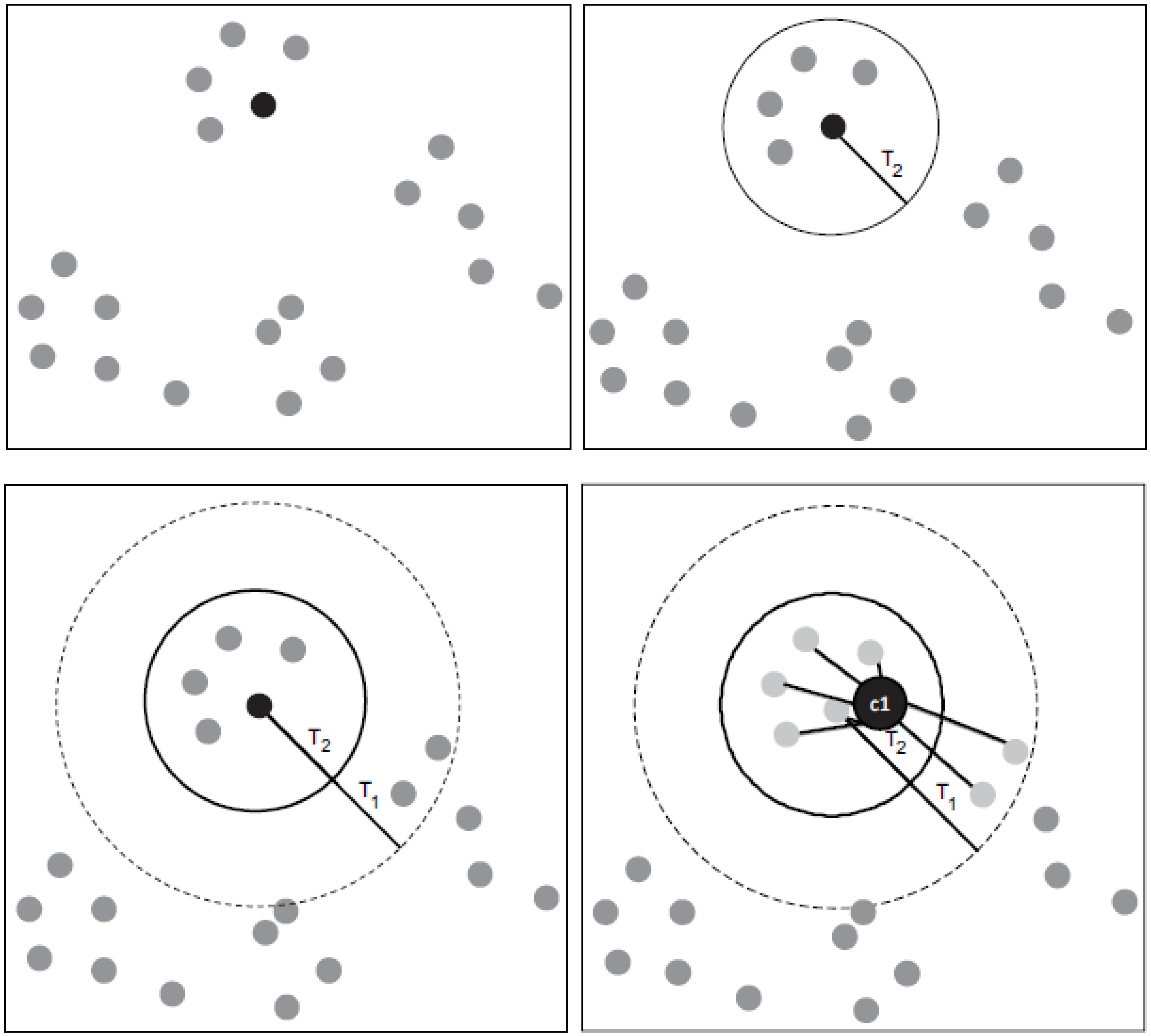
Clustering como trabajo MapReduce

```
KmeansDriver.runJob(hadoopConf,
      inputVectorFilesDirPath, clusterCenterFilesDirPath,
      outputDir, new EuclideanDistanceMeasure(),
      convergenceThreshold, numIterations, true, false);
```

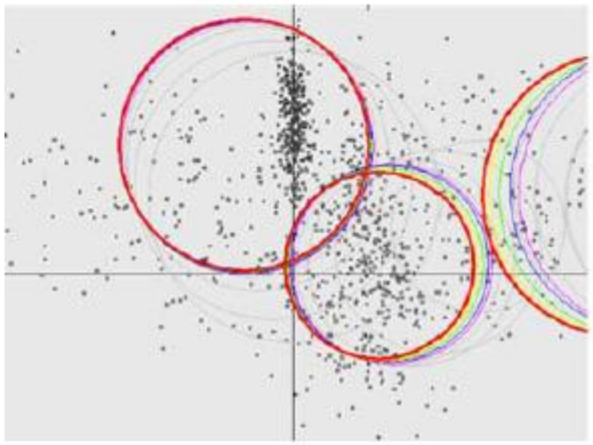
Ejecución de ejemplo:

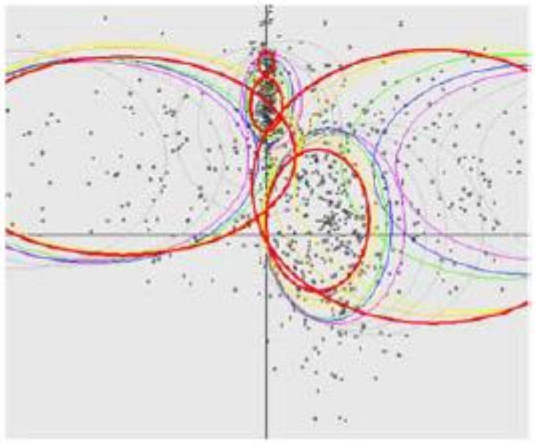
```
$ bin/mahout kmeans -i reuters-vectors/tfidf-vectors/ \
-c reuters-initial-clusters  \
-o reuters-kmeans-clusters \
-dm org.apache.mahout.common.distance.SquaredEuclideanDistanceMeasure \
-cd 1.0 -k 20 -x 20 -cl
```

Canopy Clustering

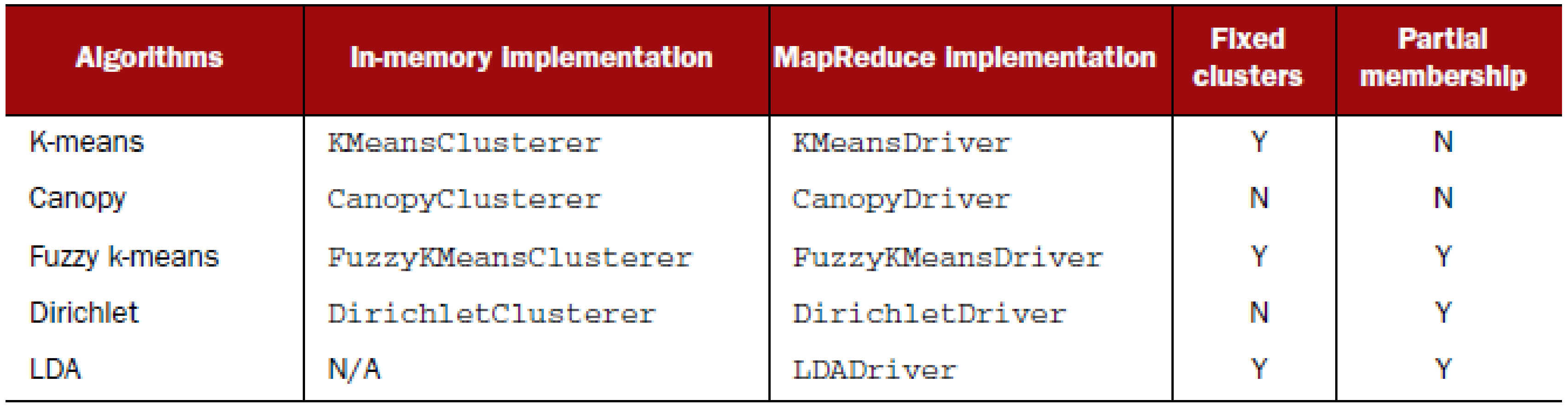


Model-based (Dirichlet) clustering





Clases para los métodos de clustering



Algorithms	In-memory implementation	MapReduce implementation	Fixed clusters	Partial membership
K-means	`KMeansClusterer`	`KMeansDriver`	Y	N
Canopy	`CanopyClusterer`	`CanopyDriver`	N	N
Fuzzy k-means	`FuzzyKMeansClusterer`	`FuzzyKMeansDriver`	Y	Y
Dirichlet	`DirichletClusterer`	`DirichletDriver`	N	Y
LDA	N/A	`LDADriver`	Y	Y

Aplicaciones reales del clustering de Mahout

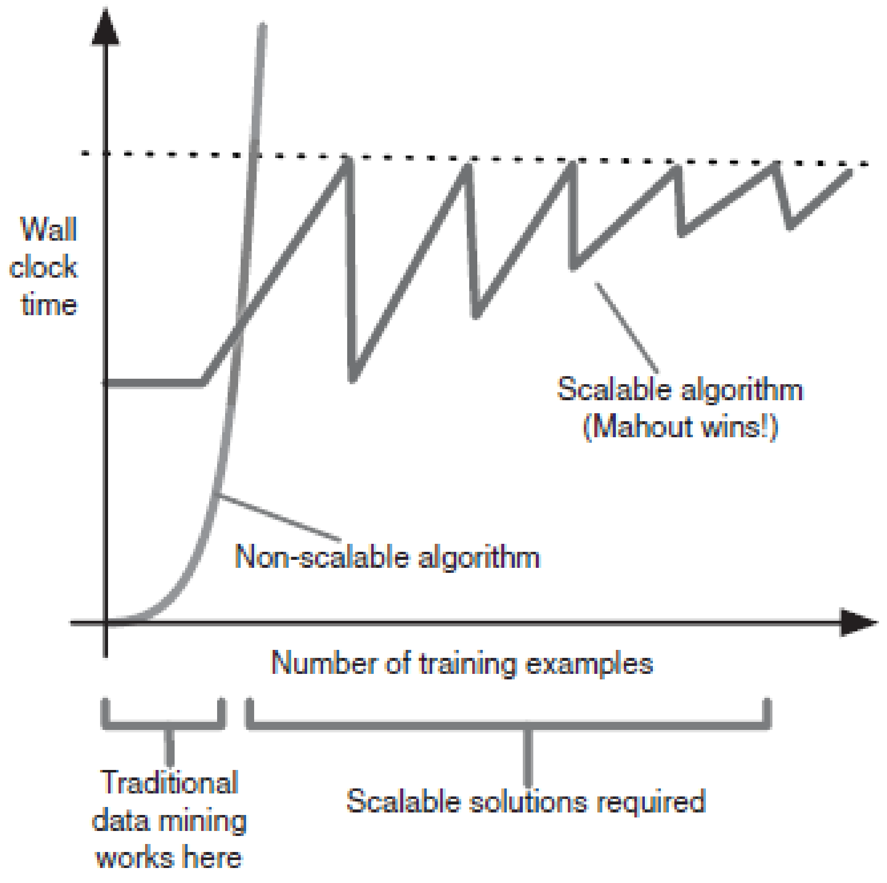
- Localizar usuarios similares en Twitter
- Sugerencia de artistas en Last.fm
- Análisis de los datos de Stack Overflow

Clasificación

Mahout en clasificación

- ¿Por qué usar Mahout en clasificación?
- La principal ventaja es su capacidad para escalar a conjuntos de datos grandes
- En general, las necesidades de recursos en Mahout crecen linealmente con el tamaño de los problemas

Escalabilidad de Mahout en clasificación



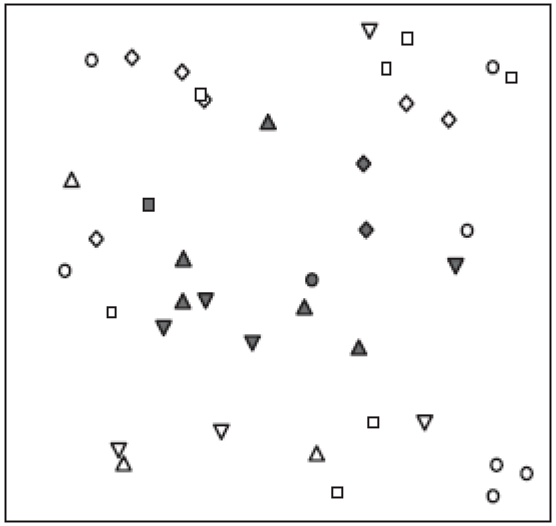
Métodos de clasificación en Mahout

- Modelos ingenuos de Bayes
- Modelos ocultos de Markov
- Regresión Logística
- Random Forests

Flujo de trabajo en un problema de clasificación

1. Crear el modelo:
    1. Salida
    2. Entradas
    3. Recoletar datos
    4. Seleccionar algoritmo de aprendizaje
    5. Entrenar el modelo
2. Evaluar el modelo
3. Usar el modelo en producción

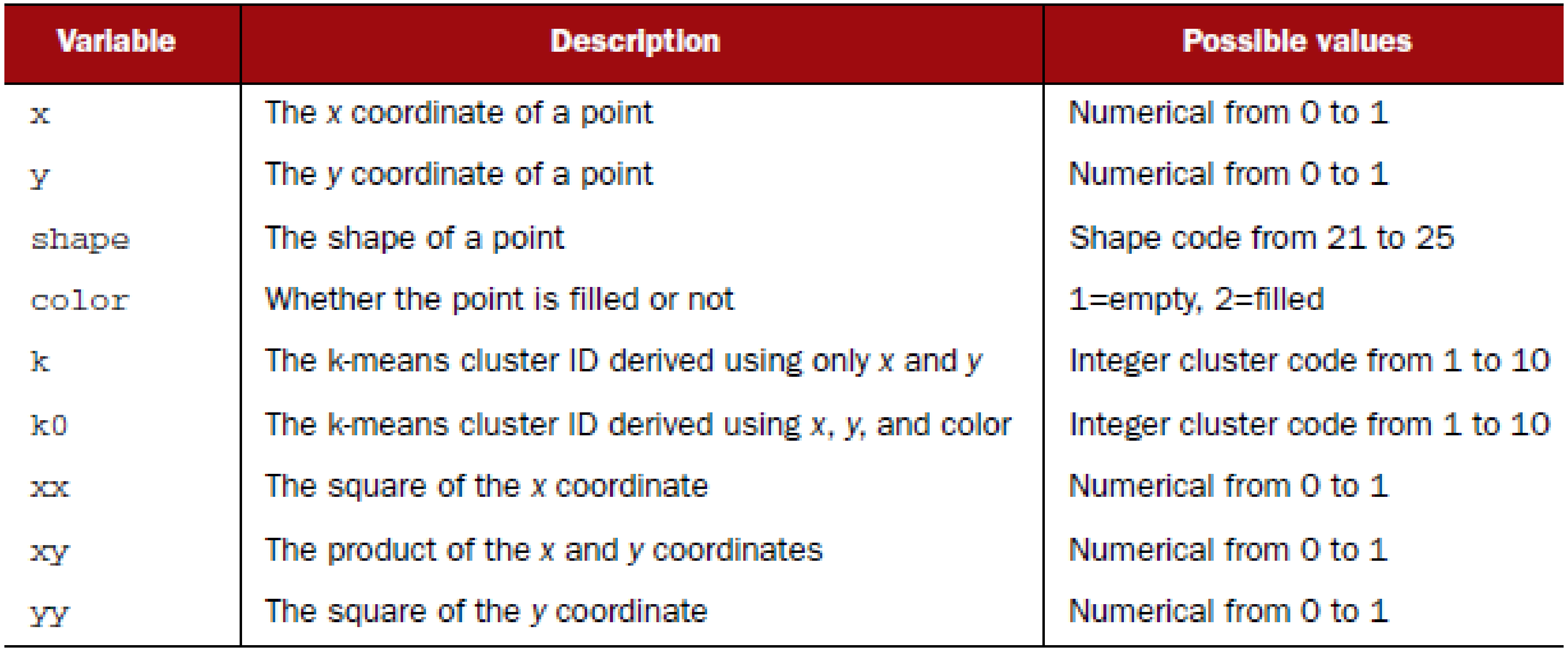
Clasificación de puntos rellenos



Distinguir los puntos que

tienen relleno de los que no

Características de los datos



Variable	Description	Possible values
`x`	The *x* coordinate of a point	Numerical from 0 to 1
`y`	The *y* coordinate of a point	Numerical from 0 to 1
`shape`	The shape of a point	Shape code from 21 to 25
`color`	Whether the point is filled or not	1=empty, 2=filled
`k`	The k-means cluster ID derived using only *x* and *y*	Integer cluster code from 1 to 10
`k0`	The k-means cluster ID derived using *x*, *y*, and color	Integer cluster code from 1 to 10
`xx`	The square of the *x* coordinate	Numerical from 0 to 1
`xy`	The product of the *x* and *y* coordinates	Numerical from 0 to 1
`yy`	The square of the *y* coordinate	Numerical from 0 to 1

¿Qué características son necesarias?

Construyendo el clasificador

```
mahout trainlogistic --input donut.csv
  --output ./model --target color --categories 2
  --predictors x y --types numeric --features 20
  --passes 100 --rate 50
```

Salida

```
MAHOUT-JOB: /usr/lib/mahout/mahout-examples-0.7-cdh4.7.0-job.jar

20

color ~ -0.149*Intercept Term + -0.701*x + -0.427*y

      Intercept Term -0.14885

                   x -0.70136

                   y -0.42740

    0.000000000     0.000000000     0.000000000     0.000000000     0.000000000
0.000000000     0.000000000     0.000000000     0.000000000     0.000000000    -
0.701362221     0.000000000     0.000000000    -0.148846792     0.000000000
0.000000000     0.000000000    -0.427403872     0.000000000     0.000000000

14/08/16 18:24:15 INFO driver.MahoutDriver: Program took 601 ms (Minutes:
0.010016666666666667)
```

Evaluar el modelo

```
mahout runlogistic --input donut.csv --model
./model --auc –confusion
```

```
AUC = 0.57

confusion: [[27.0, 13.0], [0.0, 0.0]]

entropy: [[-0.4, -0.3], [-1.2, -0.7]]

14/08/16 18:27:06 INFO driver.MahoutDriver:
Program took 127 ms (Minutes:
0.0021333333333333334)
```

Mejorando el modelo

```
mahout trainlogistic --input donut.csv --output
./model --target color --categories 2 --
predictors x y a b c  --types numeric --features
20 --passes 100 --rate 50
```

La nueva evaluación

```
mahout runlogistic --input donut.csv --model
./model --auc --confusion
```

```
AUC = 1.00

confusion: [[27.0, 1.0], [0.0, 12.0]]

entropy: [[-0.1, -1.5], [-4.0, -0.2]]
```

Sobre un conjunto de prueba

```
mahout runlogistic --input donut-test.csv --
model ./model --auc -confusion
```

```
AUC = 0.97

confusion: [[24.0, 2.0], [3.0, 11.0]]

entropy: [[-0.2, -2.8], [-4.1, -0.1]]
```

Clasificación de mensajes de USENET

- Ejemplo: Compilación de mensajes de 20 USENET newsgroups
- http://mahout.apache.org/users/classification/twenty-newsgroup.html

Ejecución sencilla

- cd $MAHOUT_HOME
- ./examples/bin/classify-20newsgroups.sh

```
Información: Standard NB Results:
=============================================
Summary
-------------------------------------------------------
Correctly Classified Instances          :      11159      99,1382%
Incorrectly Classified Instances        :         97       0,8618%
Total Classified Instances              :      11256

=======================================================
```

Ejecución detallada

1. Crear directorio de trabajo
2. Decargar y desempaquetar datos
3. Convertir los datos en SequenceFiles
4. Preprocesar los datos para incluir frecuencias de términos
5. Dividir el conjunto de datos en: entrenamiento y prueba
6. Entrenar el clasificador
7. Evaluar el clasificador

1. Crear un directorio de trabajo

```
export WORK_DIR=/tmp/mahout-work-${USER}
mkdir -p ${WORK_DIR}
```

2. Descargar y desempaquetar datos

```
curl
http://people.csail.mit.edu/jrennie/20Newsgroups/2
0news-bydate.tar.gz -o ${WORK_DIR}/20news-
bydate.tar.gz

mkdir -p ${WORK_DIR}/20news-bydate

cd ${WORK_DIR}/20news-bydate && tar xzf ../20news-
bydate.tar.gz && cd .. && cd ..

mkdir ${WORK_DIR}/20news-all

cp -R ${WORK_DIR}/20news-bydate/*/*
${WORK_DIR}/20news-all
```

2.b Si se usa Hadoop

```
hadoop dfs -put ${WORK_DIR}/20news-all
${WORK_DIR}/20news-all
```

3. Convertir los datos en SequenceFile

```
mahout seqdirectory

        -i ${WORK_DIR}/20news-all

        -o ${WORK_DIR}/20news-seq

        -ow
```

4. Preprocesar los datos: frecuencias de términos

```
mahout seq2sparse
        -i ${WORK_DIR}/20news-seq
        -o ${WORK_DIR}/20news-vectors
        -lnorm
        -nv
        -wt tfidf
```

5. Dividir el conjunto de datos

```
mahout split
        -i ${WORK_DIR}/20news-vectors/tfidf-vectors
        --trainingOutput ${WORK_DIR}/20news-train-vectors
        --testOutput ${WORK_DIR}/20news-test-vectors
        --randomSelectionPct 40
        --overwrite --sequenceFiles -xm sequential
```

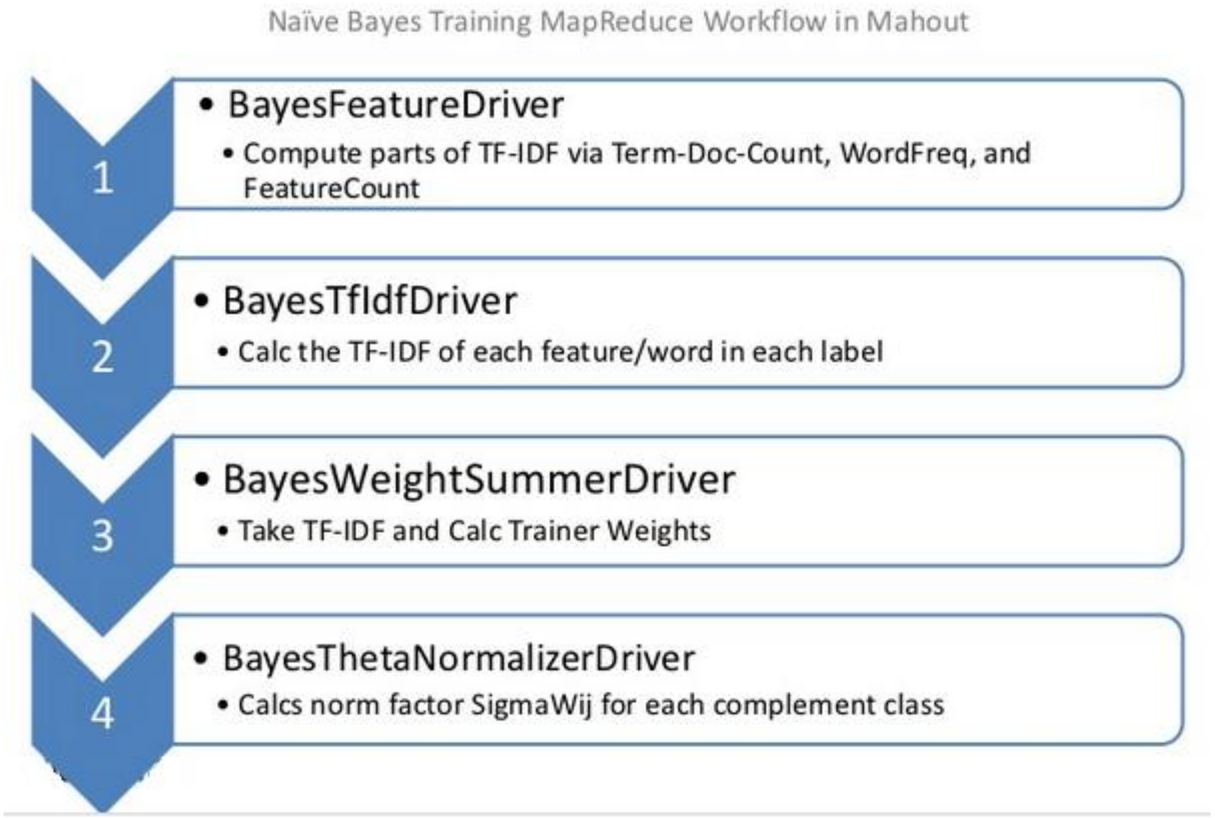
6. Entrenar el clasificador

```
mahout trainnb

        -i ${WORK_DIR}/20news-train-vectors

        -el

        -o ${WORK_DIR}/model

        -li ${WORK_DIR}/labelindex

        -ow

        -c
```

7. Evaluar el clasificador

```
mahout testnb
        -i ${WORK_DIR}/20news-test-vectors
        -m ${WORK_DIR}/model
        -l ${WORK_DIR}/labelindex
        -ow
        -o ${WORK_DIR}/20news-testing
        -c
```

Flujo de trabajo


Naïve Bayes Training MapReduce Workflow in Mahout

1
- BayesFeatureDriver
  - Compute parts of TF-IDF via Term-Doc-Count, WordFreq, and FeatureCount

2
- BayesTfIdfDriver
  - Calc the TF-IDF of each feature/word in each label

3
- BayesWeightSummerDriver
  - Take TF-IDF and Calc Trainer Weights

4
- BayesThetaNormalizerDriver
  - Calcs norm factor SigmaWij for each complement class

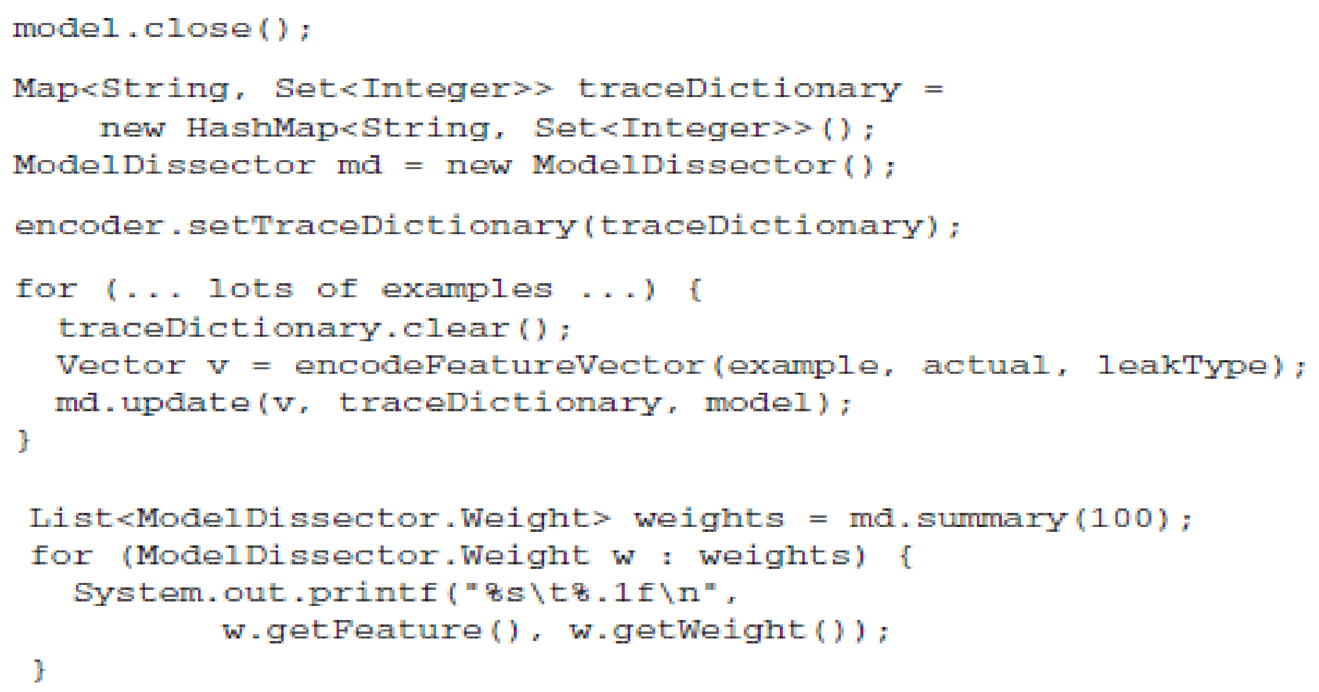
Evaluación de clasificadores

- La API de Mahout para evaluación de clasificadores:
  - Percent correct
  - Confusion matrix
  - Entropy matrix
  - AUC
  - Log Likelihood

Size of data set	Mahout algorithm	Execution model	Characteristics
Small to medium (less than tens of millions of training examples)	Stochastic gradient descent (SGD) family: `OnlineLogisticRegression`, `CrossFoldLearner`, `AdaptiveLogisticRegression`	Sequential, online, incremental	Uses all types of predictor variables; sleek and efficient over the appropriate data range (up to millions of training examples)
	Support vector machine (SVM)	Sequential	Experimental still; sleek and efficient over the appropriate data range
Medium to large (millions to hundreds of millions of training examples)	Naive Bayes	Parallel	Strongly prefers text-like data; medium to high overhead for training; effective and useful for data sets too large for SGD or SVM
	Complementary naive Bayes	Parallel	Somewhat more expensive to train than naive Bayes; effective and useful for data sets too large for SGD, but has similar limitations to naive Bayes
Small to medium (less than tens of millions of training examples)	Random forests	Parallel	Uses all types of predictor variables; high overhead for training; not widely used (yet); costly but offers complex and interesting classifications, handles nonlinear and conditional relationships in data better than other techniques

Disección de modelos

- (Mahout) Proceso para identificar el efecto de distintas entradas en la salida del clasificador



```
model.close();

Map<String, Set<Integer>> traceDictionary =
    new HashMap<String, Set<Integer>>();
ModelDissector md = new ModelDissector();

encoder.setTraceDictionary(traceDictionary);

for (... lots of examples ...) {
  traceDictionary.clear();
  Vector v = encodeFeatureVector(example, actual, leakType);
  md.update(v, traceDictionary, model);
}

 List<ModelDissector.Weight> weights = md.summary(100);
 for (ModelDissector.Weight w : weights) {
   System.out.printf("%s\t%.1f\n",
          w.getFeature(), w.getWeight());
 }
```

Diseño de sistemas para *muchos* datos

1. Determinar requisitos: tamaño y tiempo de respuesta
2. Extracción de características
3. Codificación de los datos
4. Desarrollo
5. Explotación
6. Actualización

El camino que enruta …

- Se aleja de Hadoop y busca otras plataformas con más soporte: Spark
- Interfaces para nuevos lenguajes: Scala

Conclusiones

- Es una biblioteca para ML con la escalabilidad como requisito fundamental
- Incluye implementaciones secuenciales y paralelas (basadas en MapReduce)
- Ha sido probada en exigentes aplicaciones reales
- Cambio de soporte para escalabilidad
- Biblioteca en desarrollo…

Referencias

- S. Owen, R. Anil, T. Dunning, E. Friedman, "Mahout in Action", Manning, 2011
- P. Giacomelli, "Apache Mahout Cookbook", PacktPublishing, 2013
- mahout.apache.org
- Stack Overflow