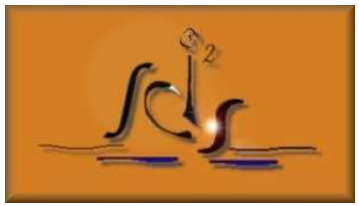
Ciencia de Datos: Un Enfoque Práctico en la Era del Big Data
Hadoop: Caso Práctico 3

Sara Del Río García

Departamento de Ciencias de la Computación e Inteligencia Artificial,
E.T.S. de Ingenierías Informática y de Telecomunicación,
srio@decsai.ugr.es

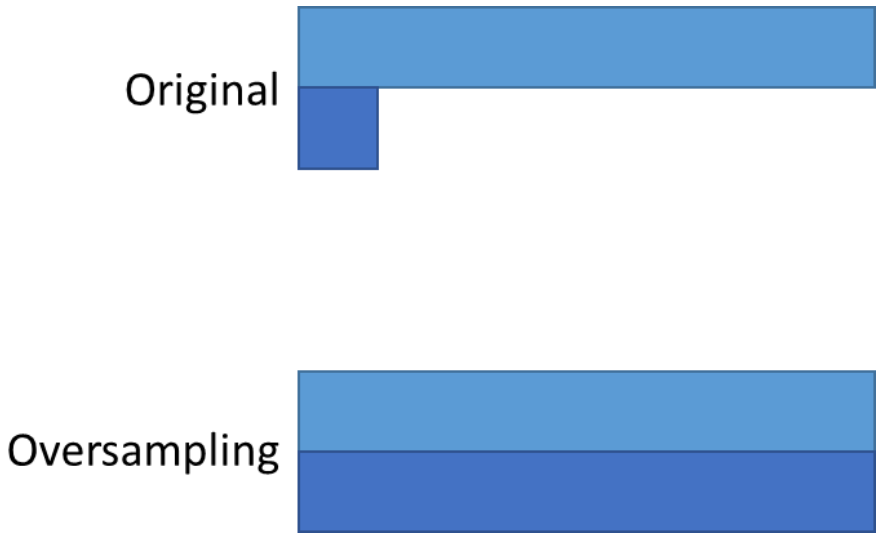


Contenido

Contenido

Ejemplo: Random Oversampling (sobremuestreo aleatorio)

- Es un método no heurístico que intenta ajustar la distribución de clases a través de la replicación aleatoria de ejemplos en la clase minoritaria.



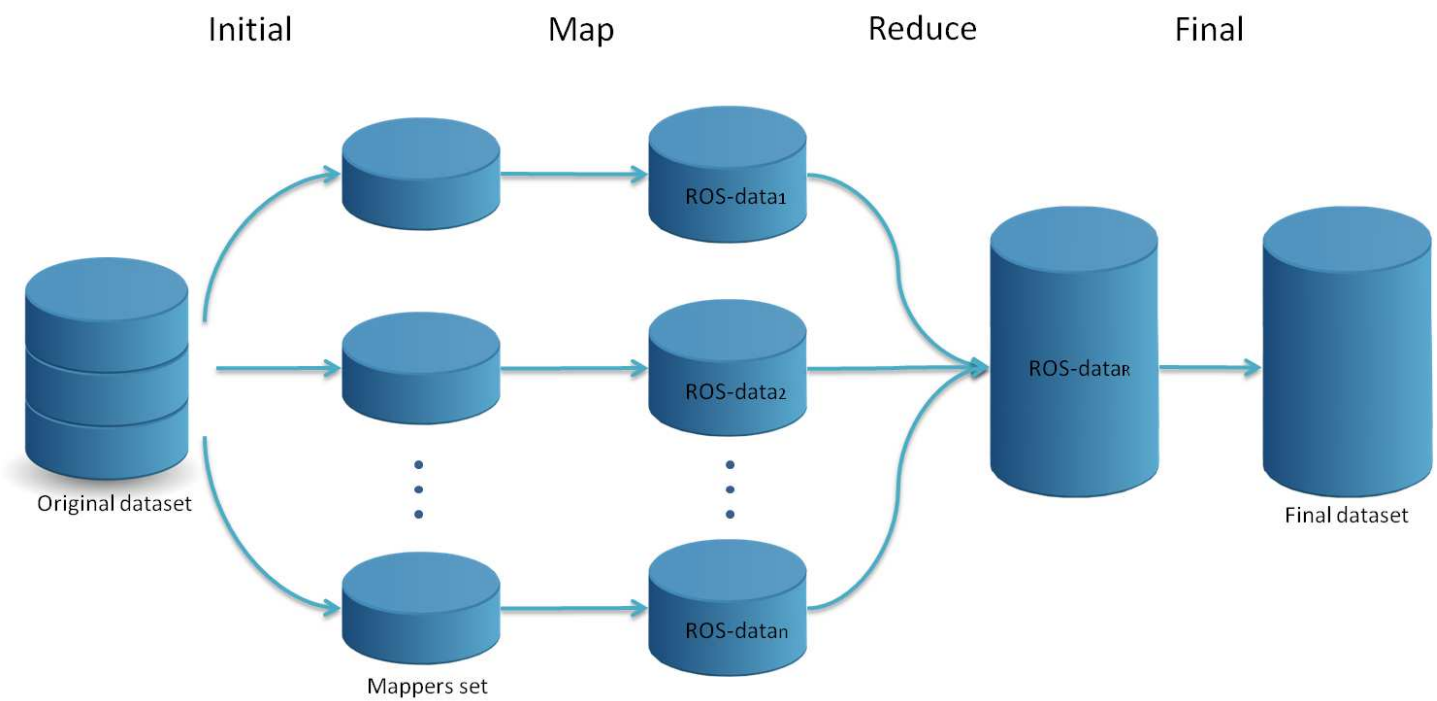
Contenido

Random Oversampling usando MapReduce (ROS-MapReduce)

- Permite obtener un conjunto de datos con una distribución balanceada de clases mediante la replicación aleatoria de ejemplos de la clase minoritaria
- Este algoritmo consta de dos fases diferentes:
  1. **Fase Map**: cada proceso map se encarga de balacear la distribución de clases de su partición mediante la replicación de instancias de la clase minoritaria
  2. **Fase Reduce**: se combinan cada una de las salidas de los maps para formar el conjunto de datos final ya balanceado

Random Oversampling usando MapReduce (ROS-MapReduce)

- Diagrama de flujo de ROS-MapReduce



Random Oversampling usando MapReduce (ROS-MapReduce) - Map Class

```
    public void map(LongWritable key, Text value, Context
        context) throws IOException, InterruptedException {
Instance instance = converter.convert(value.toString());
int label_code = (int)dataset.getLabel(instance);
String label = dataset.getLabelString(label_code);
LongWritable id;
Random r = new Random();
if (!noOutput) {
    if(label.equalsIgnoreCase(negativeClass)){
        int random = r.nextInt(replication);
        id = new LongWritable(random);
        context.write(id, value);
    }
    else{
        for(int i = 0 ; i < replication ; i++){
            id = new LongWritable(i);
            context.write(id, value);
        }
    }
}
}
```

Random Oversampling usando MapReduce (ROS-MapReduce) - Reduce Class

```
        public void reduce(LongWritable key, Iterable<Text>
            values, Context context) throws IOException,
            InterruptedException {
    NullWritable id = null;

    for (Text value : values) {
      context.write(id, value);
    }
  }
```

Random Oversampling usando MapReduce (ROS-MapReduce)

- Marco Experimental:
  - Caso de estudio derivado del conjunto de datos **Iris**
    - # Características: 4
    - # Instancias: 60
    - # Clases: 2 {Iris-setosa, Iris-versicolor}
    - % Iris-setosa: 33.33, % Iris-versicolor: 66.67
  - Disponible en *UCI Machine Learning Repository*
  - Esquema de validación cruzada en 5 particiones (usaremos la primera partición)

Random Oversampling usando MapReduce (ROS-MapReduce)

- Caso de estudio: 5 maps
- Uso:
  1. Generar el fichero que describe al conjunto de datos:

```
hadoop jar mahout-distribution-cm.jar
org.apache.mahout.classifier.df.tools.Describe
-p datasets/iris/iris-imbalanced-5-1tra.arff
-f datasets/iris/iris-imbalanced-5-1tra.info
-d 4 N L
```

Random Oversampling usando MapReduce (ROS-MapReduce)

- Caso de estudio: 5 maps
- Uso:
  2. Calcular el tamaño del split (por ejemplo, 5 maps):

```
FILE_SIZE=( ‘hadoop fs -ls
datasets/iris/iris-imbalanced-5-1tra.arff | awk ’{print $5}’‘)
BYTES_BY_PARTITION=$((FILE_SIZE/5))
MAX_BYTES_BY_PARTITION=$((BYTES_BY_PARTITION+1))
```

Random Oversampling usando MapReduce (ROS-MapReduce)

- Caso de estudio: 5 maps
- Uso:
  3. Calcular el número de instancias de la clase minoritaria o menos representativa. En este ejemplo, "Iris-setosa" es el nombre la clase minoritaria:

     NPOS=(‘hadoop fs -cat datasets/iris/iris-imbalanced-5-1tra.arff | grep ’,Iris-setosa$’ | wc -l‘)

Random Oversampling usando MapReduce (ROS-MapReduce)

- Caso de estudio: 5 maps
- Uso:
  4. Calcular el número de instancias de la clase mayoritaria. En este ejemplo, "Iris-versicolor" es el nombre la clase mayoritaria:

     NNEG=(‘hadoop fs -cat datasets/iris/iris-imbalanced-5-1tra.arff | grep ’,Iris-versicolor$’ | wc -l‘)

Random Oversampling usando MapReduce (ROS-MapReduce)

- Caso de estudio: 5 maps
- Uso:
  5. Ejecutar Random Oversampling (por ejemplo, 5 maps):
     hadoop jar mahout-distribution-cm.jar

     org.apache.mahout.classifier.df.mapreduce.Resampling
     -Dmapreduce.input.fileinputformat.split.minsize=
     $BYTES_BY_PARTITION
     -Dmapreduce.input.fileinputformat.split.maxsize=
     $MAX_BYTES_BY_PARTITION
     -dp datasets/iris/iris-imbalanced-5-1tra.arff
     -d output-ROS-iris
     -ds datasets/iris/iris-imbalanced-5-1tra.info
     -rs overs -p 5 -npos $NPOS -nneg $NNEG
     -negclass Iris-versicolor -tm ROS-iris-build_time

Random Oversampling usando MapReduce (ROS-MapReduce)

- Caso de estudio: 5 maps
- Uso:
  6. Comprobar la salida. El fichero de salida tendrá el nombre "part-r-00000".

     hadoop fs -ls output-ROS-iris

Comprobar el estado de las ejecuciones a través de la siguiente consola web:

http://localhost:8088/cluster

Contenido

Referencias

- Apache Mahout:

  http://mahout.apache.org/

- **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. doi: 10.1016/j.ins.2014.03.043
  http://sci2s.ugr.es/sites/default/files/ficherosPublicaciones/1742_2014-delRio-INS.pdf

- UCI Machine Learning Repository - Iris dataset:

  https://archive.ics.uci.edu/ml/datasets/Iris

Happy Hadooping!

Ciencia de Datos: Un Enfoque Práctico en la Era del Big Data
Hadoop: Caso Práctico 3

Sara Del Río García

Departamento de Ciencias de la Computación e Inteligencia Artificial,
E.T.S. de Ingenierías Informática y de Telecomunicación,
srio@decsai.ugr.es