
UNIVERSIDAD DE GRANADA

MASTER PROFESIONAL EN INGENIERÍA INFORMÁTICA

PRÁCTICA 2

Hadoop

Autor:

Manuel Jesús García Manday
(nickter@correo.ugr.es)

Master en Ingeniería Informática

20 de mayo de 2017

Índice

1. Objetivo.	3
2. Introducción.	3
3. Calcula el valor mínimo de la variable (columna) 5.	4
4. Calcula el valor máximo de la variable (columna) 5.	6
5. Calcula al mismo tiempo los valores máximo y mínimo de la variable 5.	9
6. Calcula los valores máximo y mínimo de todas las variables (salvo la última, que es la etiqueta de la clase).	12
7. Realizar la media de la variable 5.	12
8. Obtener la media de todas las variables (salvo la clase).	12
9. Comprobar si el conjunto de datos ECBDL es balanceado o no balanceado, es decir, que el ratio entre clases sea menor o mayor que 1.5 respectivamente.	12
10.Cálculo del coeficiente de correlación entre todas las parejas de variables.	12

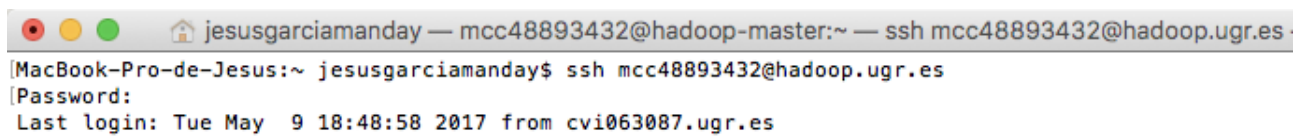
1. Objetivo.

El objetivo de esta práctica es realizar programas escalables para mejorar la eficiencia en entornos Big Data.

2. Introducción.

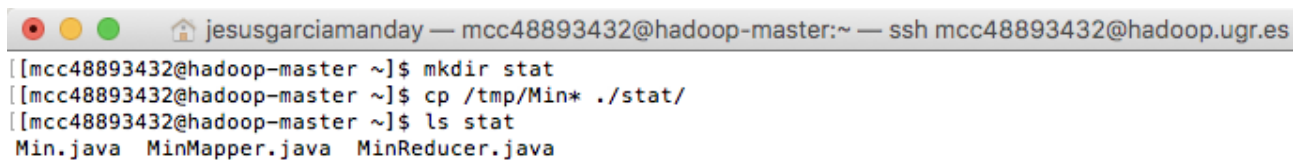
Para comenzar a realizar las tareas que se piden en esta práctica, es necesario en primer lugar realizar una serie de pasos iniciales que se describen a continuación.

Realizamos una conexión remota hacia el servidor **hadoop.ugr.es** y una vez dentro creamos una carpeta nueva donde descargaremos el código Java de los programas. Comprobamos también que los datos de entrada se encuentran disponibles.



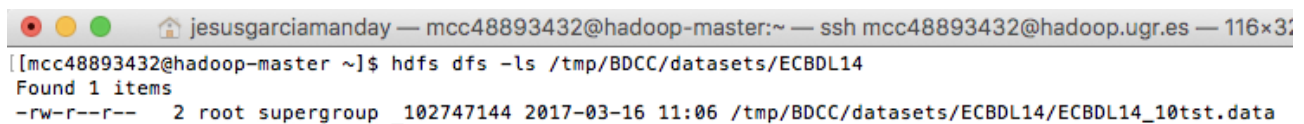
```
jesusgarciamanday — mcc48893432@hadoop-master:~ — ssh mcc48893432@hadoop.ugr.es
[MacBook-Pro-de-Jesus:~ jesusgarciamanday$ ssh mcc48893432@hadoop.ugr.es
[Password:
Last login: Tue May  9 18:48:58 2017 from cvi063087.ugr.es
```

Figura 1: Conexión remota a **hadoop.ugr.es**.



```
jesusgarciamanday — mcc48893432@hadoop-master:~ — ssh mcc48893432@hadoop.ugr.es
[mcc48893432@hadoop-master ~]$ mkdir stat
[mcc48893432@hadoop-master ~]$ cp /tmp/Min* ./stat/
[mcc48893432@hadoop-master ~]$ ls stat
Min.java  MinMapper.java  MinReducer.java
```

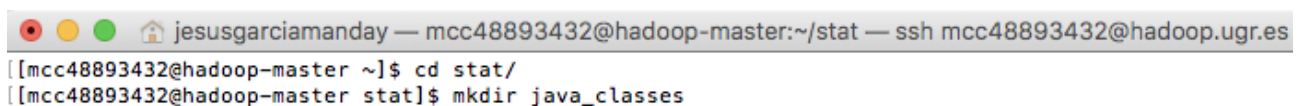
Figura 2: Copiando los ficheros Java.



```
jesusgarciamanday — mcc48893432@hadoop-master:~ — ssh mcc48893432@hadoop.ugr.es — 116x3
[mcc48893432@hadoop-master ~]$ hdfs dfs -ls /tmp/BDCC/datasets/ECBDL14
Found 1 items
-rw-r--r--  2 root supergroup _102747144 2017-03-16 11:06 /tmp/BDCC/datasets/ECBDL14/ECBDL14_10tst.data
```

Figura 3: Datos de entrada.

Viendo que están disponibles, ahora nos creamos un directorio local para las clases de java.



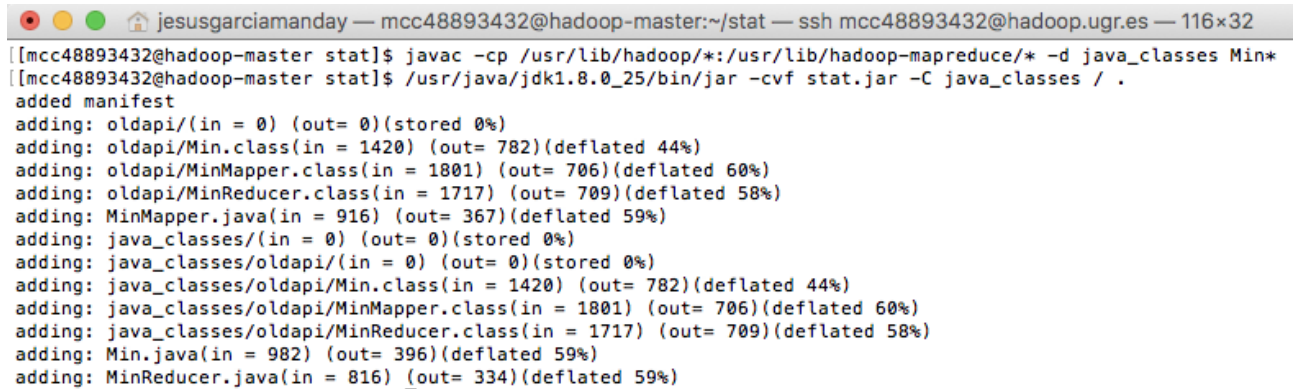
```
jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es
[mcc48893432@hadoop-master ~]$ cd stat/
[mcc48893432@hadoop-master stat]$ mkdir java_classes
```

Figura 4: Directorio local.

Con la preconfiguración realizada pasamos a realizar las diferentes tareas que se exponen en la práctica.

3. Calcula el valor mínimo de la variable (columna) 5.

La primera de ellas es calcular el mínimo sobre el conjunto de valores del dataset, por lo que una vez que tenemos todos los ficheros java correspondientes, ahora toca compilarlos para crear el fichero **.jar** a continuación y ejecutarlo en **hadoop**.

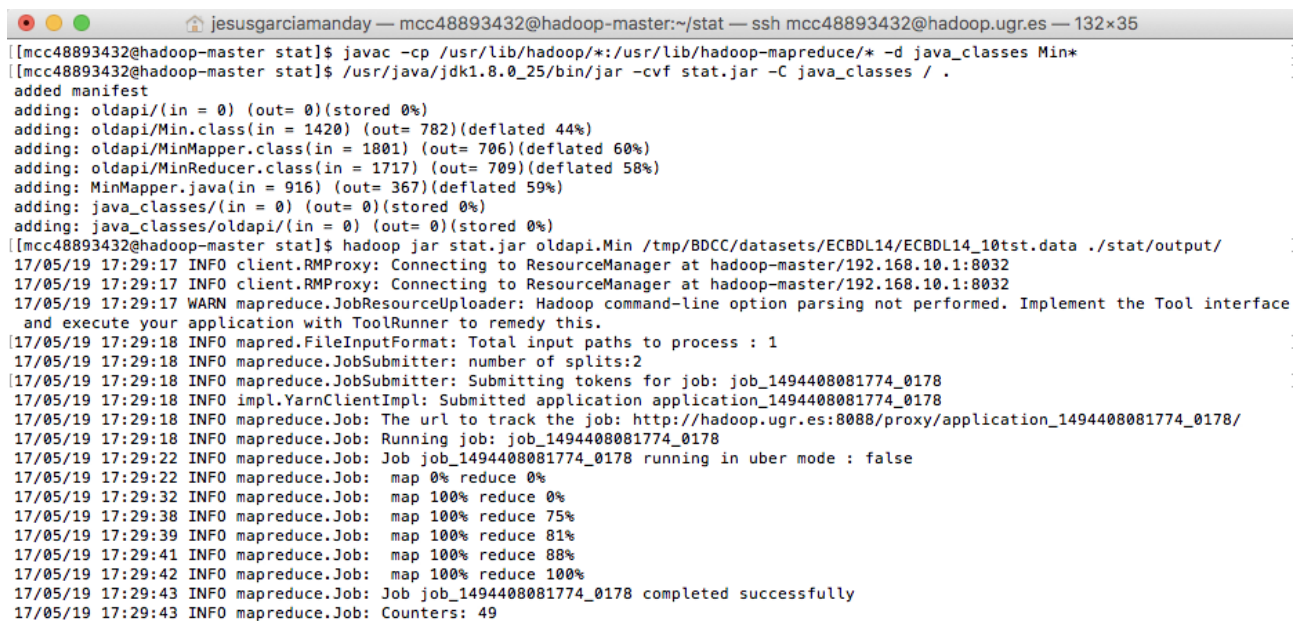


```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 116x32
[mcc48893432@hadoop-master stat]$ javac -cp /usr/lib/hadoop/*:/usr/lib/hadoop-mapreduce/* -d java_classes Min*
[mcc48893432@hadoop-master stat]$ /usr/java/jdk1.8.0_25/bin/jar -cvf stat.jar -C java_classes / .
added manifest
adding: oldapi/(in = 0) (out= 0)(stored 0%)
adding: oldapi/Min.class(in = 1420) (out= 782)(deflated 44%)
adding: oldapi/MinMapper.class(in = 1801) (out= 706)(deflated 60%)
adding: oldapi/MinReducer.class(in = 1717) (out= 709)(deflated 58%)
adding: MinMapper.java(in = 916) (out= 367)(deflated 59%)
adding: java_classes/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/Min.class(in = 1420) (out= 782)(deflated 44%)
adding: java_classes/oldapi/MinMapper.class(in = 1801) (out= 706)(deflated 60%)
adding: java_classes/oldapi/MinReducer.class(in = 1717) (out= 709)(deflated 58%)
adding: Min.java(in = 982) (out= 396)(deflated 59%)
adding: MinReducer.java(in = 816) (out= 334)(deflated 59%)

```

Figura 5: Compilamos y ejecutamos (I).



```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 132x35
[mcc48893432@hadoop-master stat]$ javac -cp /usr/lib/hadoop/*:/usr/lib/hadoop-mapreduce/* -d java_classes Min*
[mcc48893432@hadoop-master stat]$ /usr/java/jdk1.8.0_25/bin/jar -cvf stat.jar -C java_classes / .
added manifest
adding: oldapi/(in = 0) (out= 0)(stored 0%)
adding: oldapi/Min.class(in = 1420) (out= 782)(deflated 44%)
adding: oldapi/MinMapper.class(in = 1801) (out= 706)(deflated 60%)
adding: oldapi/MinReducer.class(in = 1717) (out= 709)(deflated 58%)
adding: MinMapper.java(in = 916) (out= 367)(deflated 59%)
adding: java_classes/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/(in = 0) (out= 0)(stored 0%)
[mcc48893432@hadoop-master stat]$ hadoop jar stat.jar oldapi.Min /tmp/BDCC/datasets/ECBDL14/ECBDL14_10tst.data ./stat/output/
17/05/19 17:29:17 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/19 17:29:17 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/19 17:29:17 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface
and execute your application with ToolRunner to remedy this.
17/05/19 17:29:18 INFO mapred.FileInputFormat: Total input paths to process : 1
17/05/19 17:29:18 INFO mapreduce.JobSubmitter: number of splits:2
17/05/19 17:29:18 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1494408081774_0178
17/05/19 17:29:18 INFO impl.YarnClientImpl: Submitted application application_1494408081774_0178
17/05/19 17:29:18 INFO mapreduce.Job: The url to track the job: http://hadoop.ugr.es:8088/proxy/application_1494408081774_0178/
17/05/19 17:29:18 INFO mapreduce.Job: Running job: job_1494408081774_0178
17/05/19 17:29:22 INFO mapreduce.Job: Job job_1494408081774_0178 running in uber mode : false
17/05/19 17:29:22 INFO mapreduce.Job: map 0% reduce 0%
17/05/19 17:29:32 INFO mapreduce.Job: map 100% reduce 0%
17/05/19 17:29:38 INFO mapreduce.Job: map 100% reduce 75%
17/05/19 17:29:39 INFO mapreduce.Job: map 100% reduce 81%
17/05/19 17:29:41 INFO mapreduce.Job: map 100% reduce 88%
17/05/19 17:29:42 INFO mapreduce.Job: map 100% reduce 100%
17/05/19 17:29:43 INFO mapreduce.Job: Job job_1494408081774_0178 completed successfully
17/05/19 17:29:43 INFO mapreduce.Job: Counters: 49

```

Figura 6: Compilamos y ejecutamos (II).

```

File System Counters
  FILE: Number of bytes read=2142847
  FILE: Number of bytes written=6470142
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=102749934
  HDFS: Number of bytes written=8
  HDFS: Number of read operations=54
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=32
Job Counters
  Launched map tasks=2
  Launched reduce tasks=16
  Rack-local map tasks=2
  Total time spent by all maps in occupied slots (ms)=112574
  Total time spent by all reduces in occupied slots (ms)=1894977
  Total time spent by all map tasks (ms)=16082
  Total time spent by all reduce tasks (ms)=38673
  Total vcore-seconds taken by all map tasks=16082
  Total vcore-seconds taken by all reduce tasks=38673
  Total megabyte-seconds taken by all map tasks=112574000
  Total megabyte-seconds taken by all reduce tasks=1933650000

```

Figura 7: Compilamos y ejecutamos (III).


```

Map-Reduce Framework
  Map input records=2897917
  Map output records=2897917
  Map output bytes=28979170
  Map output materialized bytes=2143005
  Input split bytes=234
  Combine input records=0
  Combine output records=0
  Reduce input groups=1
  Reduce shuffle bytes=2143005
  Reduce input records=2897917
  Reduce output records=1
  Spilled Records=5795834
  Shuffled Maps =32
  Failed Shuffles=0
  Merged Map outputs=32
  GC time elapsed (ms)=347
  CPU time spent (ms)=37010
  Physical memory (bytes) snapshot=7926947840
  Virtual memory (bytes) snapshot=984134000640
  Total committed heap usage (bytes)=19421724672
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=102749700
File Output Format Counters
  Bytes Written=8

```

Figura 8: Compilamos y ejecutamos (IV).

Por último comprobamos el resultado para ver si se ha realizado correctamente.



```

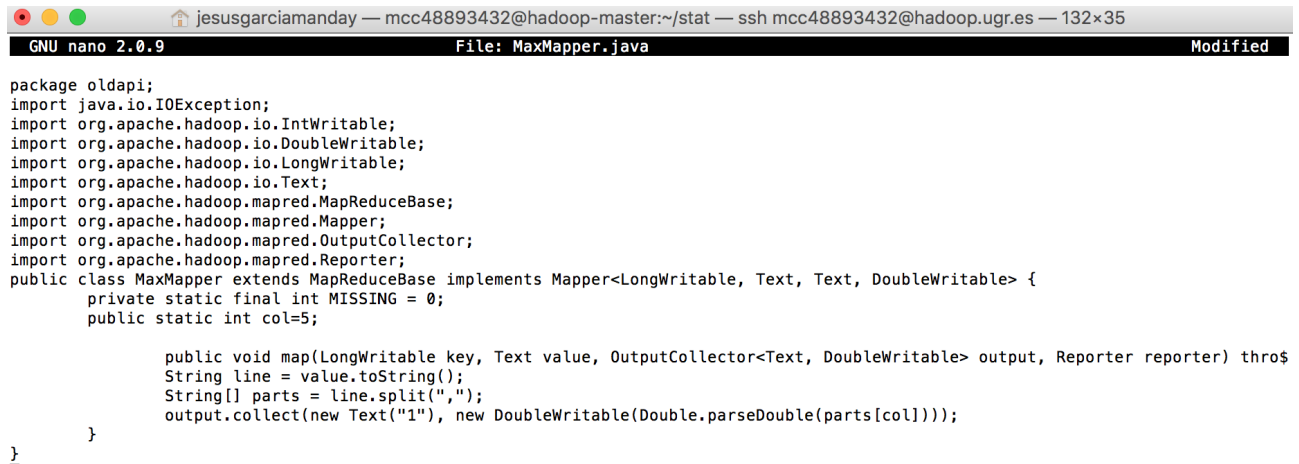
jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es
[mcc48893432@hadoop-master stat]$ hdfs dfs -cat stat/output/*
1 -11.0

```

Figura 9: Comprobando resultado.

4. Calcula el valor máximo de la variable (columna) 5.

Para calcular el valor máximo de la variable (columna) 5 vamos a crear clases de java correspondientes para hacer dicha operación. Comenzaremos por crear la clase **MaxMapper** con su fichero java correspondiente.



```

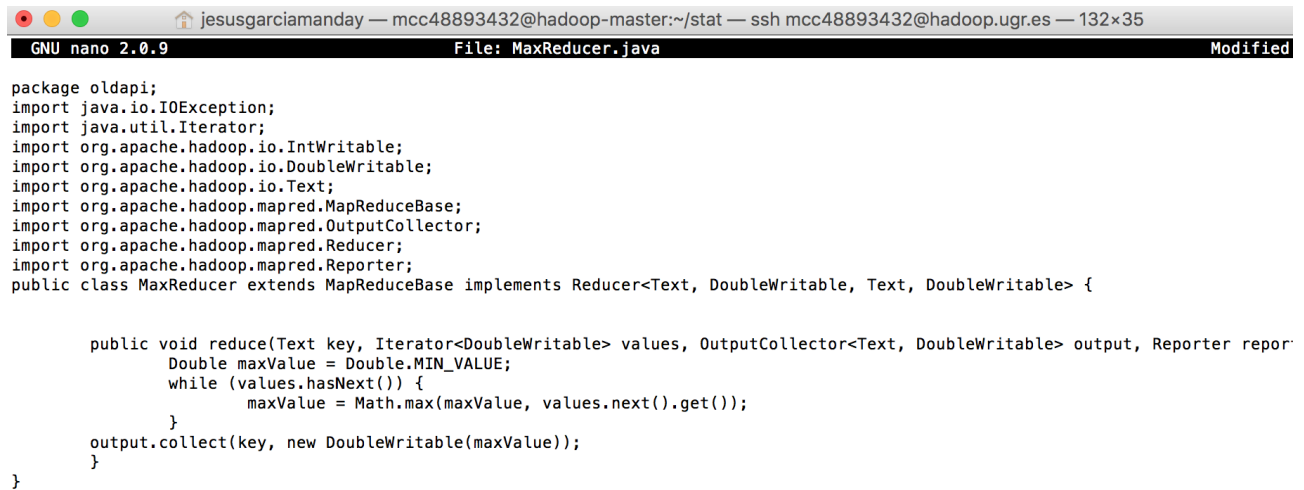
package oldapi;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class MaxMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, DoubleWritable> {
    private static final int MISSING = 0;
    public static int col=5;

    public void map(LongWritable key, Text value, OutputCollector<Text, DoubleWritable> output, Reporter reporter) thro$
        String line = value.toString();
        String[] parts = line.split(",");
        output.collect(new Text("1"), new DoubleWritable(Double.parseDouble(parts[col])));
    }
}

```

Figura 10: Clase MaxMapper.

A continuación creamos la clase correspondientes para el Reducer (**MaxReducer**) y la clase principal **Max** donde estará el main.



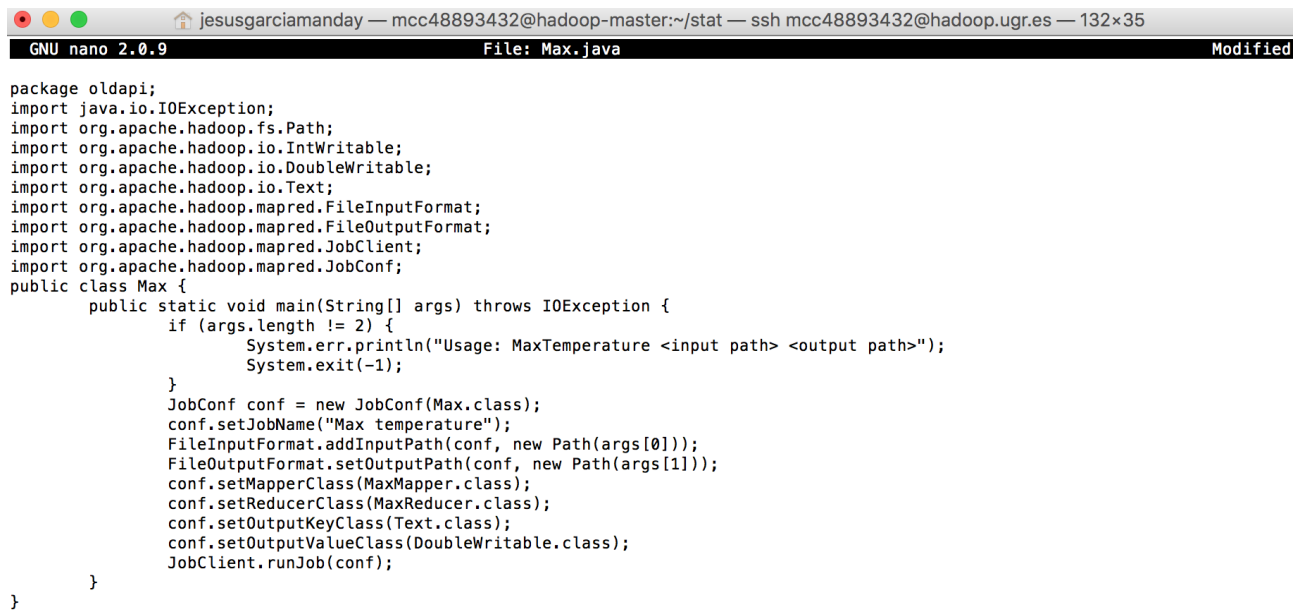
```

package oldapi;
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
public class MaxReducer extends MapReduceBase implements Reducer<Text, DoubleWritable, Text, DoubleWritable> {

    public void reduce(Text key, Iterator<DoubleWritable> values, OutputCollector<Text, DoubleWritable> output, Reporter repor
        Double maxVal = Double.MIN_VALUE;
        while (values.hasNext()) {
            maxVal = Math.max(maxVal, values.next().get());
        }
        output.collect(key, new DoubleWritable(maxVal));
    }
}

```

Figura 11: Clase MaxReducer.



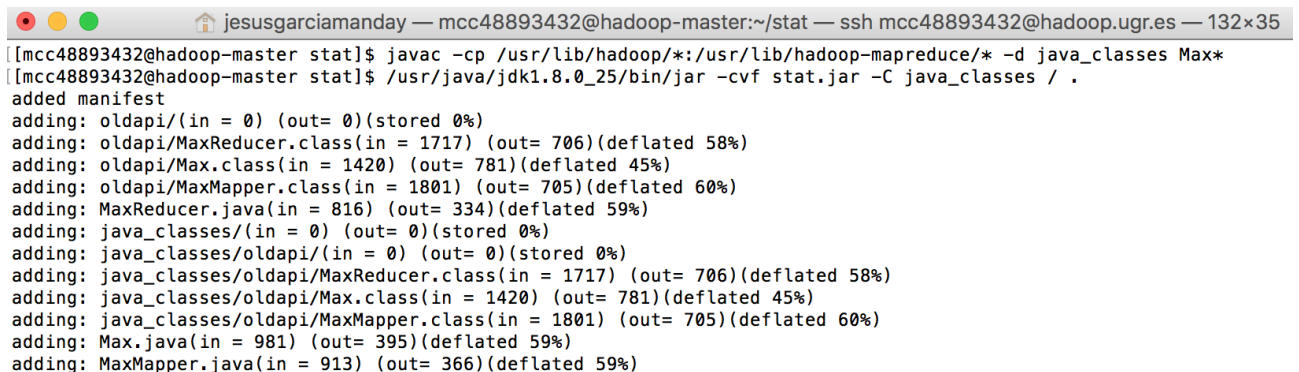
```

package oldapi;
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
public class Max {
    public static void main(String[] args) throws IOException {
        if (args.length != 2) {
            System.err.println("Usage: MaxTemperature <input path> <output path>");
            System.exit(-1);
        }
        JobConf conf = new JobConf(Max.class);
        conf.setJobName("Max temperature");
        FileInputFormat.addInputPath(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(MaxMapper.class);
        conf.setReducerClass(MaxReducer.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(DoubleWritable.class);
        JobClient.runJob(conf);
    }
}

```

Figura 12: Clase Max.

Una vez que tenemos todos los ficheros procedemos a realizar los mismos pasos que con el ejercicio anterior compilando las clases, creando el fichero jar y ejecutandolo en **hadoop**.



```

[[mcc48893432@hadoop-master stat]$ javac -cp /usr/lib/hadoop-*/usr/lib/hadoop-mapreduce/* -d java_classes Max*
[[mcc48893432@hadoop-master stat]$ /usr/java/jdk1.8.0_25/bin/jar -cvf stat.jar -C java_classes / .
added manifest
adding: oldapi/(in = 0) (out= 0)(stored 0%)
adding: oldapi/MaxReducer.class(in = 1717) (out= 706)(deflated 58%)
adding: oldapi/Max.class(in = 1420) (out= 781)(deflated 45%)
adding: oldapi/MaxMapper.class(in = 1801) (out= 705)(deflated 60%)
adding: MaxReducer.java(in = 816) (out= 334)(deflated 59%)
adding: java_classes/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/MaxReducer.class(in = 1717) (out= 706)(deflated 58%)
adding: java_classes/oldapi/Max.class(in = 1420) (out= 781)(deflated 45%)
adding: java_classes/oldapi/MaxMapper.class(in = 1801) (out= 705)(deflated 60%)
adding: Max.java(in = 981) (out= 395)(deflated 59%)
adding: MaxMapper.java(in = 913) (out= 366)(deflated 59%)

```

Figura 13: Compilamos y ejecutamos (I).


```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 132x35
[mcc48893432@hadoop-master stat]$ hadoop jar stat.jar oldapi.Max /tmp/BDCC/datasets/ECBDL14/ECBDL14_10tst.data ./stat/output2/
17/05/20 12:57:54 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 12:57:54 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 12:57:55 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface
and execute your application with ToolRunner to remedy this.
17/05/20 12:57:55 INFO mapred.FileInputFormat: Total input paths to process : 1
17/05/20 12:57:55 INFO mapreduce.JobSubmitter: number of splits:2
17/05/20 12:57:55 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1494408081774_0246
17/05/20 12:57:55 INFO impl.YarnClientImpl: Submitted application application_1494408081774_0246
17/05/20 12:57:55 INFO mapreduce.Job: The url to track the job: http://hadoop.ugr.es:8088/proxy/application_1494408081774_0246/
17/05/20 12:57:55 INFO mapreduce.Job: Running job: job_1494408081774_0246
17/05/20 12:58:01 INFO mapreduce.Job: Job job_1494408081774_0246 running in uber mode : false
17/05/20 12:58:01 INFO mapreduce.Job: map 0% reduce 0%
17/05/20 12:58:11 INFO mapreduce.Job: map 100% reduce 0%
17/05/20 12:58:16 INFO mapreduce.Job: map 100% reduce 81%
17/05/20 12:58:19 INFO mapreduce.Job: map 100% reduce 88%
17/05/20 12:58:20 INFO mapreduce.Job: map 100% reduce 100%
17/05/20 12:58:20 INFO mapreduce.Job: Job job_1494408081774_0246 completed successfully
17/05/20 12:58:20 INFO mapreduce.Job: Counters: 49
    File System Counters
      FILE: Number of bytes read=2171063
      FILE: Number of bytes written=6526311
      FILE: Number of read operations=0
      FILE: Number of large read operations=0
      FILE: Number of write operations=0
      HDFS: Number of bytes read=102749934
      HDFS: Number of bytes written=6
      HDFS: Number of read operations=54
      HDFS: Number of large read operations=0
      HDFS: Number of write operations=32

```

Figura 14: Compilamos y ejecutamos (II).

```

Job Counters
  Launched map tasks=2
  Launched reduce tasks=16
  Rack-local map tasks=2
  Total time spent by all maps in occupied slots (ms)=111125
  Total time spent by all reduces in occupied slots (ms)=1915900
  Total time spent by all map tasks (ms)=15875
  Total time spent by all reduce tasks (ms)=39100
  Total vcore-seconds taken by all map tasks=15875
  Total vcore-seconds taken by all reduce tasks=39100
  Total megabyte-seconds taken by all map tasks=111125000
  Total megabyte-seconds taken by all reduce tasks=1955000000

Map-Reduce Framework
  Map input records=2897917
  Map output records=2897917
  Map output bytes=28979170
  Map output materialized bytes=2170940
  Input split bytes=234
  Combine input records=0
  Combine output records=0
  Reduce input groups=1
  Reduce shuffle bytes=2170940
  Reduce input records=2897917
  Reduce output records=1
  Spilled Records=5795834
  Shuffled Maps =32
  Failed Shuffles=0
  Merged Map outputs=32
  GC time elapsed (ms)=327
  CPU time spent (ms)=36200
  Physical memory (bytes) snapshot=7933112320
  Virtual memory (bytes) snapshot=984107524096
  Total committed heap usage (bytes)=19421724672

```

Figura 15: Compilamos y ejecutamos (III).


```

Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=102749700
File Output Format Counters
  Bytes Written=6

```

Figura 16: Compilamos y ejecutamos (IV).

Comprobamos que el resultado nos arroja el valor máximo.



```

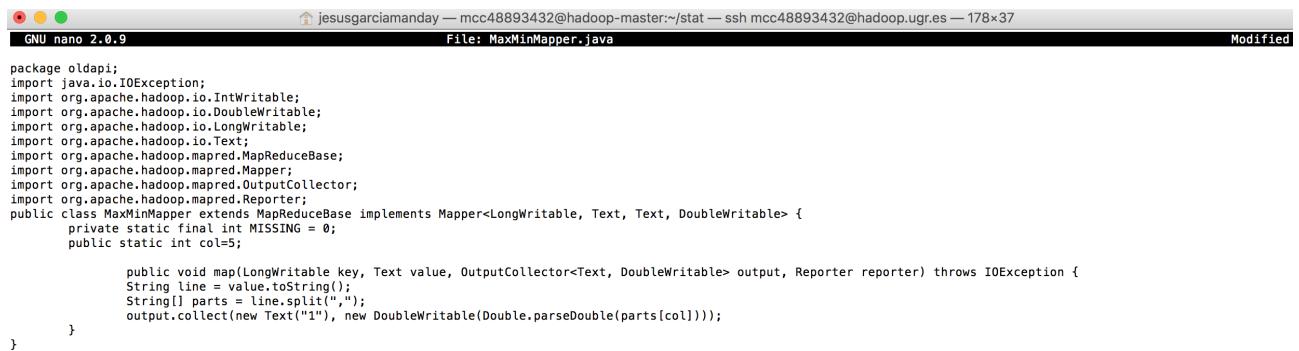
jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es
[[mcc48893432@hadoop-master stat]$ hdfs dfs -cat stat/output2/*
1      9.0

```

Figura 17: Comprobando resultado.

5. Calcula al mismo tiempo los valores máximo y mínimo de la variable 5.

Al igual que para los anteriores cálculos creamos los ficheros correspondientes, aunque el fichero con la función **Mapper** y el principal con el main solo cambia el nombre, en la clase con la función **Reducer** es donde se realizan los cambios necesarios para obtener los resultados esperados.



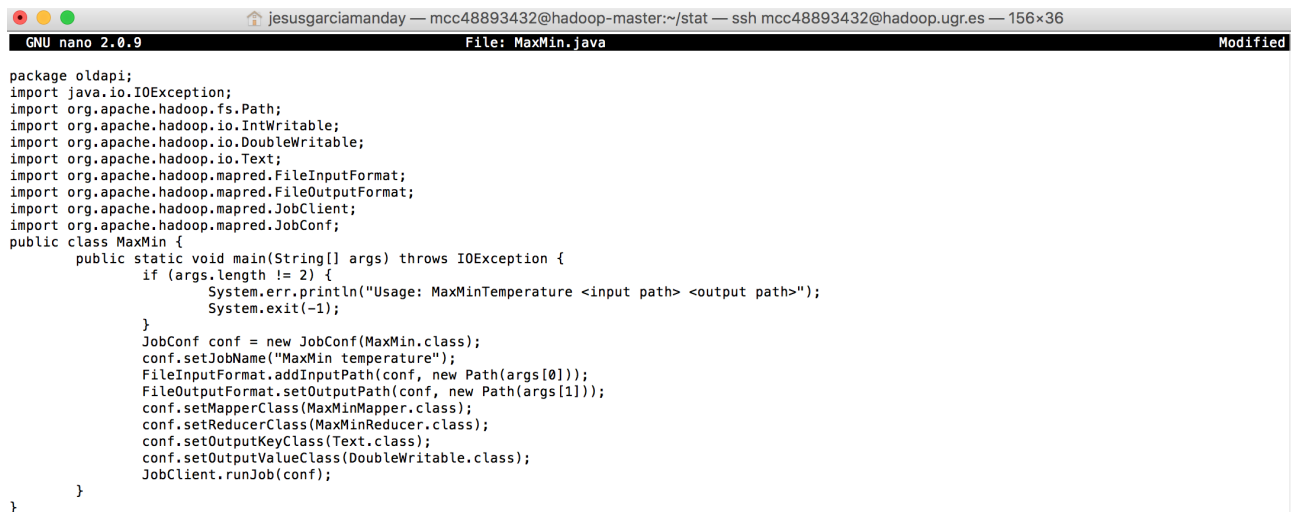
```

GNU nano 2.0.9                               File: MaxMinMapper.java                               Modified
package oldapi;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class MaxMinMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, DoubleWritable> {
    private static final int MISSING = 0;
    public static int col=5;

    public void map(LongWritable key, Text value, OutputCollector<Text, DoubleWritable> output, Reporter reporter) throws IOException {
        String line = value.toString();
        String[] parts = line.split(",");
        output.collect(new Text("1"), new DoubleWritable(Double.parseDouble(parts[col])));
    }
}

```

Figura 18: Clase MaxMinMapper.

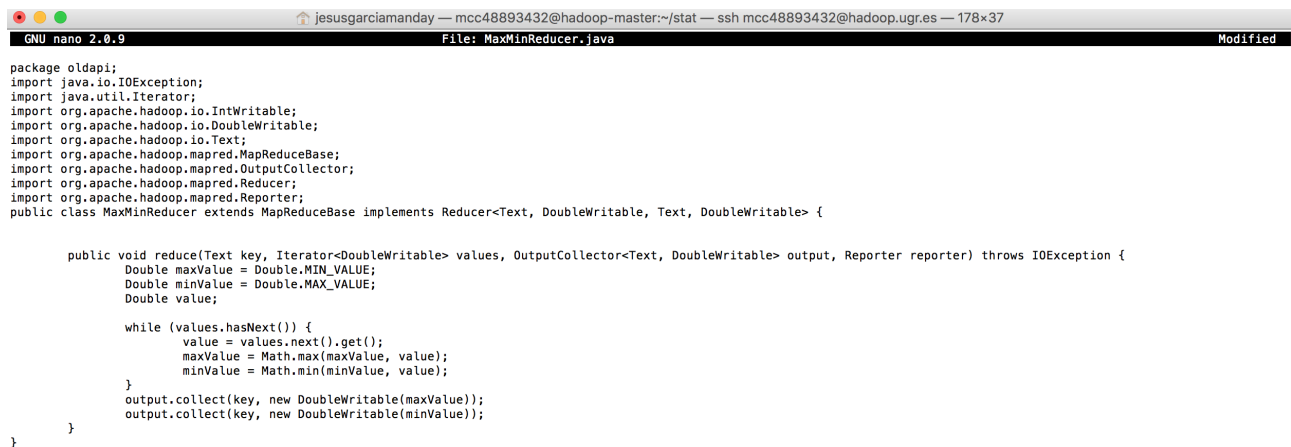


```

GNU nano 2.0.9 File: MaxMin.java Modified
package oldapi;
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
public class MaxMin {
    public static void main(String[] args) throws IOException {
        if (args.length != 2) {
            System.err.println("Usage: MaxMinTemperature <input path> <output path>");
            System.exit(-1);
        }
        JobConf conf = new JobConf(MaxMin.class);
        conf.setJobName("MaxMin temperature");
        FileInputFormat.addInputPath(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(MaxMinMapper.class);
        conf.setReducerClass(MaxMinReducer.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(DoubleWritable.class);
        JobClient.runJob(conf);
    }
}

```

Figura 19: Clase MaxMin.



```

GNU nano 2.0.9 File: MaxMinReducer.java Modified
package oldapi;
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;
public class MaxMinReducer extends MapReduceBase implements Reducer<Text, DoubleWritable, Text, DoubleWritable> {

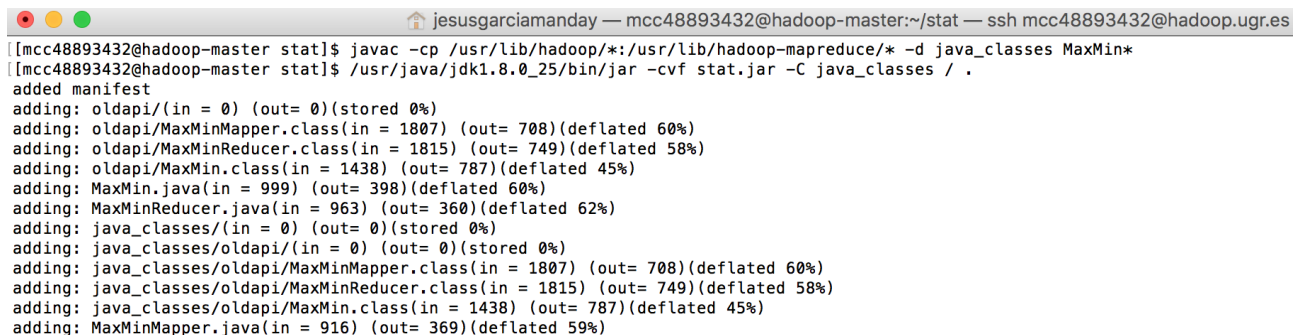
    public void reduce(Text key, Iterator<DoubleWritable> values, OutputCollector<Text, DoubleWritable> output, Reporter reporter) throws IOException {
        Double maxValue = Double.MIN_VALUE;
        Double minValue = Double.MAX_VALUE;
        Double value;

        while (values.hasNext()) {
            value = values.next().get();
            maxValue = Math.max(maxValue, value);
            minValue = Math.min(minValue, value);
        }
        output.collect(key, new DoubleWritable(maxValue));
        output.collect(key, new DoubleWritable(minValue));
    }
}

```

Figura 20: Clase MaxMinReducer.

Ahora toca compilarlos para adjuntarlos en el fichero **jar** y realizar la ejecución en **hadoop**.



```

[mcc48893432@hadoop-master stat]$ javac -cp /usr/lib/hadoop-*/usr/lib/hadoop-mapreduce/* -d java_classes MaxMin*
[mcc48893432@hadoop-master stat]$ /usr/java/jdk1.8.0_25/bin/jar -cvf stat.jar -C java_classes / .
added manifest
adding: oldapi/(in = 0) (out= 0)(stored 0%)
adding: oldapi/MaxMinMapper.class(in = 1807) (out= 708)(deflated 60%)
adding: oldapi/MaxMinReducer.class(in = 1815) (out= 749)(deflated 58%)
adding: oldapi/MaxMin.class(in = 1438) (out= 787)(deflated 45%)
adding: MaxMin.java(in = 999) (out= 398)(deflated 60%)
adding: MaxMinReducer.java(in = 963) (out= 360)(deflated 62%)
adding: java_classes/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/MaxMinMapper.class(in = 1807) (out= 708)(deflated 60%)
adding: java_classes/oldapi/MaxMinReducer.class(in = 1815) (out= 749)(deflated 58%)
adding: java_classes/oldapi/MaxMin.class(in = 1438) (out= 787)(deflated 45%)
adding: MaxMinMapper.java(in = 916) (out= 369)(deflated 59%)

```

Figura 21: Compilamos y ejecutamos (I).

```

[mcc48893432@hadoop-master stat]$ hadoop jar stat.jar oldapi.MaxMin /tmp/BDC/datasets/ECBDL14/ECBDL14_10tst.data ./stat/output5/
17/05/20 17:03:15 INFO client.RMProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 17:03:15 INFO client.RMProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 17:03:15 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner
to remedy this.
17/05/20 17:03:15 INFO mapred.FileInputFormat: Total input paths to process : 1
17/05/20 17:03:16 INFO mapreduce.JobSubmitter: number of splits:2
17/05/20 17:03:16 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1494408081774_0267
17/05/20 17:03:16 INFO impl.YarnClientImpl: Submitted application application_1494408081774_0267
17/05/20 17:03:16 INFO mapreduce.Job: The url to track the job: http://hadoop.ugr.es:8088/proxy/application_1494408081774_0267/
17/05/20 17:03:16 INFO mapreduce.Job: Running job: job_1494408081774_0267
17/05/20 17:03:20 INFO mapreduce.Job: Job job_1494408081774_0267 running in uber mode : false
17/05/20 17:03:20 INFO mapreduce.Job: map 0% reduce 0%
17/05/20 17:03:30 INFO mapreduce.Job: map 100% reduce 0%
17/05/20 17:03:35 INFO mapreduce.Job: map 100% reduce 63%
17/05/20 17:03:36 INFO mapreduce.Job: map 100% reduce 81%
17/05/20 17:03:38 INFO mapreduce.Job: map 100% reduce 88%
17/05/20 17:03:40 INFO mapreduce.Job: map 100% reduce 100%
17/05/20 17:03:40 INFO mapreduce.Job: Job job_1494408081774_0267 completed successfully
17/05/20 17:03:40 INFO mapreduce.Job: Counters: 50
    File System Counters
      FILE: Number of bytes read=2121182
      FILE: Number of bytes written=6427147
      FILE: Number of read operations=0
      FILE: Number of large read operations=0
      FILE: Number of write operations=0
      HDFS: Number of bytes read=102749934
      HDFS: Number of bytes written=14
      HDFS: Number of read operations=54
      HDFS: Number of large read operations=0
      HDFS: Number of write operations=32

```

Figura 22: Compilamos y ejecutamos (II).

```

Job Counters
  Launched map tasks=2
  Launched reduce tasks=16
  Data-local map tasks=1
  Rack-local map tasks=1
  Total time spent by all maps in occupied slots (ms)=111153
  Total time spent by all reduces in occupied slots (ms)=1901494
  Total time spent by all map tasks (ms)=15879
  Total time spent by all reduce tasks (ms)=38806
  Total vcore-seconds taken by all map tasks=15879
  Total vcore-seconds taken by all reduce tasks=38806
  Total megabyte-seconds taken by all map tasks=111153000
  Total megabyte-seconds taken by all reduce tasks=1940300000
Map-Reduce Framework
  Map input records=2897917
  Map output records=2897917
  Map output bytes=28979170
  Map output materialized bytes=2121495
  Input split bytes=234
  Combine input records=0
  Combine output records=0
  Reduce input groups=1
  Reduce shuffle bytes=2121495
  Reduce input records=2897917
  Reduce output records=2
  Spilled Records=5795834
  Shuffled Maps =32
  Failed Shuffles=0
  Merged Map outputs=32
  GC time elapsed (ms)=367
  CPU time spent (ms)=36140
  Physical memory (bytes) snapshot=7914434560
  Virtual memory (bytes) snapshot=984147337216
  Total committed heap usage (bytes)=19421724672

```

Figura 23: Compilamos y ejecutamos (III).

```

Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=102749700
File Output Format Counters
  Bytes Written=14

```

Figura 24: Compilamos y ejecutamos (IV).

Por último comprobamos el resultado obtenido.

```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es
[mcc48893432@hadoop-master stat]$ hdfs dfs -cat stat/output5/*
1      9.0
1     -11.0

```

Figura 25: Comprobando resultado.

6. Calcula los valores máximo y mínimo de todas las variables (salvo la última, que es la etiqueta de la clase).

Para este cálculo solo necesitaremos adaptar la clase con la función **Mapper** para cada columna ya que el **Reducer** es el mismo. Ahora la función **Mapper** asignará diferentes key, uno por cada columna.

```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 132x35
GNU nano 2.0.9 File: MaxMinMapper.java Modified
package oldapi;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class MaxMinMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, DoubleWritable> {
    private static final int MISSING = 0;
    public static int col=5;

    public void map(LongWritable key, Text value, OutputCollector<Text, DoubleWritable> output, Reporter reporter) throws
    String line = value.toString();
    String[] parts = line.split(",");
    for(int i = 1; i < parts.length; i++){
        output.collect(new Text(String.valueOf(i)), new DoubleWritable(Double.parseDouble(parts[i-1])));
    }
}

```

Figura 26: Clase MaxMinMapper.

Como en los anteriores ejercicios, compilamos y ejecutamos las clases.

```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 132x35
[mcc48893432@hadoop-master stat]$ javac -cp /usr/lib/hadoop/*:/usr/lib/hadoop-mapreduce/* -d java_classes MaxMin*
[mcc48893432@hadoop-master stat]$ /usr/java/jdk1.8.0_25/bin/jar -cvf stat.jar -C java_classes / .
added manifest
adding: oldapi/(in = 0) (out= 0)(stored 0%)
adding: oldapi/MaxMinMapper.class(in = 1938) (out= 790)(deflated 59%)
adding: oldapi/MaxMinReducer.class(in = 1824) (out= 762)(deflated 58%)
adding: oldapi/MaxMin.class(in = 1438) (out= 787)(deflated 45%)
adding: MaxMin.java(in = 999) (out= 398)(deflated 60%)
adding: MaxMinReducer.java(in = 986) (out= 372)(deflated 62%)
adding: java_classes/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/MaxMinMapper.class(in = 1938) (out= 790)(deflated 59%)
adding: java_classes/oldapi/MaxMinReducer.class(in = 1824) (out= 762)(deflated 58%)
adding: java_classes/oldapi/MaxMin.class(in = 1438) (out= 787)(deflated 45%)
adding: MaxMinMapper.java(in = 964) (out= 410)(deflated 57%)

```

Figura 27: Compilamos y ejecutamos (I).

```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 136x41
[mcc48893432@hadoop-master stat]$ hadoop jar stat.jar oldapi.MaxMin /tmp/BDCC/datasets/ECBDL14/ECBDL14_10tst.data ./stat/output6/
17/05/20 17:46:17 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 17:46:17 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 17:46:17 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and
execute your application with ToolRunner to remedy this.
17/05/20 17:46:17 INFO mapred.FileInputFormat: Total input paths to process : 1
17/05/20 17:46:18 INFO mapreduce.JobSubmitter: number of splits:2
17/05/20 17:46:18 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1494408081774_0283
17/05/20 17:46:18 INFO impl.YarnClientImpl: Submitted application application_1494408081774_0283
17/05/20 17:46:18 INFO mapreduce.Job: The url to track the job: http://hadoop.ugr.es:8088/proxy/application_1494408081774_0283/
17/05/20 17:46:18 INFO mapreduce.Job: Running job: job_1494408081774_0283
17/05/20 17:46:22 INFO mapreduce.Job: Job job_1494408081774_0283 running in uber mode : false
17/05/20 17:46:22 INFO mapreduce.Job: map 0% reduce 0%
17/05/20 17:46:32 INFO mapreduce.Job: map 32% reduce 0%
17/05/20 17:46:35 INFO mapreduce.Job: map 45% reduce 0%
17/05/20 17:46:38 INFO mapreduce.Job: map 58% reduce 0%
17/05/20 17:46:41 INFO mapreduce.Job: map 65% reduce 0%
17/05/20 17:46:44 INFO mapreduce.Job: map 71% reduce 0%
17/05/20 17:46:47 INFO mapreduce.Job: map 79% reduce 0%
17/05/20 17:46:49 INFO mapreduce.Job: map 89% reduce 0%
17/05/20 17:46:50 INFO mapreduce.Job: map 92% reduce 0%
17/05/20 17:46:51 INFO mapreduce.Job: map 100% reduce 0%
17/05/20 17:46:54 INFO mapreduce.Job: map 100% reduce 25%
17/05/20 17:46:55 INFO mapreduce.Job: map 100% reduce 31%
17/05/20 17:46:57 INFO mapreduce.Job: map 100% reduce 81%
17/05/20 17:46:58 INFO mapreduce.Job: map 100% reduce 100%
17/05/20 17:46:59 INFO mapreduce.Job: Job job_1494408081774_0283 completed successfully
17/05/20 17:46:59 INFO mapreduce.Job: Counters: 50
File System Counters
  FILE: Number of bytes read=45717534
  FILE: Number of bytes written=70501121
  FILE: Number of read operations=0
  FILE: Number of large read operations=0
  FILE: Number of write operations=0
  HDFS: Number of bytes read=102749934
  HDFS: Number of bytes written=146
  HDFS: Number of read operations=54
  HDFS: Number of large read operations=0
  HDFS: Number of write operations=32

```

Figura 28: Compilamos y ejecutamos (II).

```

Job Counters
  Launched map tasks=2
  Launched reduce tasks=16
  Data-local map tasks=1
  Rack-local map tasks=1
  Total time spent by all maps in occupied slots (ms)=368760
  Total time spent by all reduces in occupied slots (ms)=3107090
  Total time spent by all map tasks (ms)=52680
  Total time spent by all reduce tasks (ms)=63410
  Total vcore-seconds taken by all map tasks=52680
  Total vcore-seconds taken by all reduce tasks=63410
  Total megabyte-seconds taken by all map tasks=368760000
  Total megabyte-seconds taken by all reduce tasks=3170500000
Map-Reduce Framework
  Map input records=2897917
  Map output records=28979170
  Map output bytes=292689617
  Map output materialized bytes=22770570
  Input split bytes=234
  Combine input records=0
  Combine output records=0
  Reduce input groups=10
  Reduce shuffle bytes=22770570
  Reduce input records=28979170
  Reduce output records=20
  Spilled Records=86937510
  Shuffled Maps =32
  Failed Shuffles=0
  Merged Map outputs=32
  GC time elapsed (ms)=438
  CPU time spent (ms)=126880
  Physical memory (bytes) snapshot=10873622528
  Virtual memory (bytes) snapshot=984140087296
  Total committed heap usage (bytes)=21354250240
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=102749700
File Output Format Counters
  Bytes Written=146

```

Figura 29: Compilamos y ejecutamos (III).

Vemos el resultado que se ha obtenido.



```

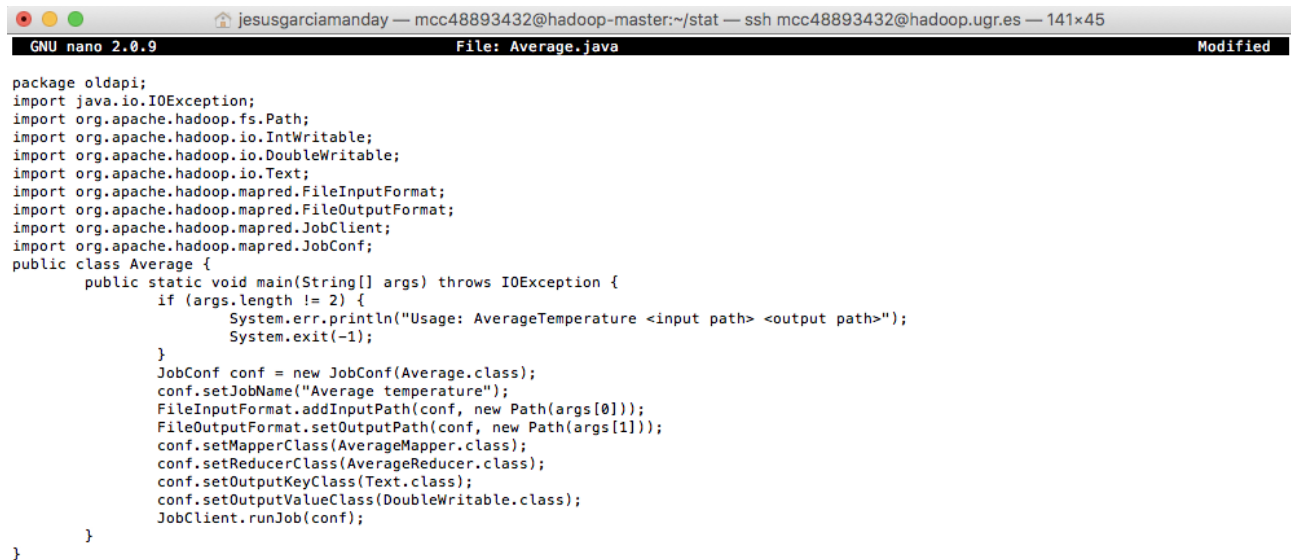
[mcc48893432@hadoop-master stat]$ hdfs dfs -cat stat/output6/*
1      0.768
1      0.094
10     10.0
10     -13.0
2      0.154
2      0.0
3      10.0
3      -12.0
4      8.0
4      -11.0
5      9.0
5      -12.0
6      9.0
6      -11.0
7      9.0
7      -13.0
8      9.0
8      -12.0
9      7.0
9      -12.0

```

Figura 30: Comprobando resultado.

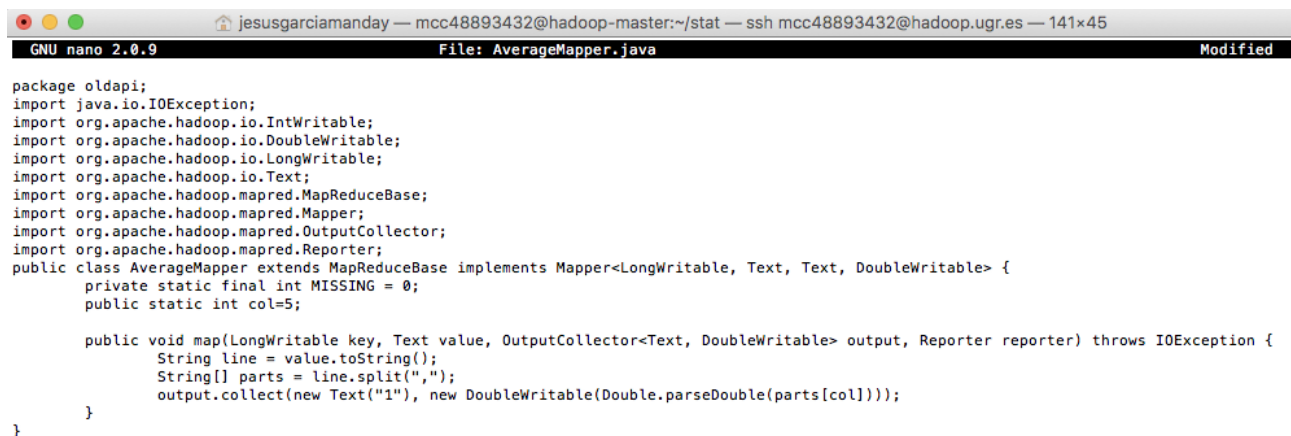
7. Realizar la media de la variable 5.

Para realizar esta tarea será necesario crear dos funciones nuevas para **Mapper** y **Reducer** con sus respectivas clases ya que difieren de las tareas anteriores.



```
package oldapi;
import java.io.IOException;
import org.apache.hadoop.fs.Path;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.FileInputFormat;
import org.apache.hadoop.mapred.FileOutputFormat;
import org.apache.hadoop.mapred.JobClient;
import org.apache.hadoop.mapred.JobConf;
public class Average {
    public static void main(String[] args) throws IOException {
        if (args.length != 2) {
            System.err.println("Usage: AverageTemperature <input path> <output path>");
            System.exit(-1);
        }
        JobConf conf = new JobConf(Average.class);
        conf.setJobName("Average temperature");
        FileInputFormat.addInputPath(conf, new Path(args[0]));
        FileOutputFormat.setOutputPath(conf, new Path(args[1]));
        conf.setMapperClass(AverageMapper.class);
        conf.setReducerClass(AverageReducer.class);
        conf.setOutputKeyClass(Text.class);
        conf.setOutputValueClass(DoubleWritable.class);
        JobClient.runJob(conf);
    }
}
```

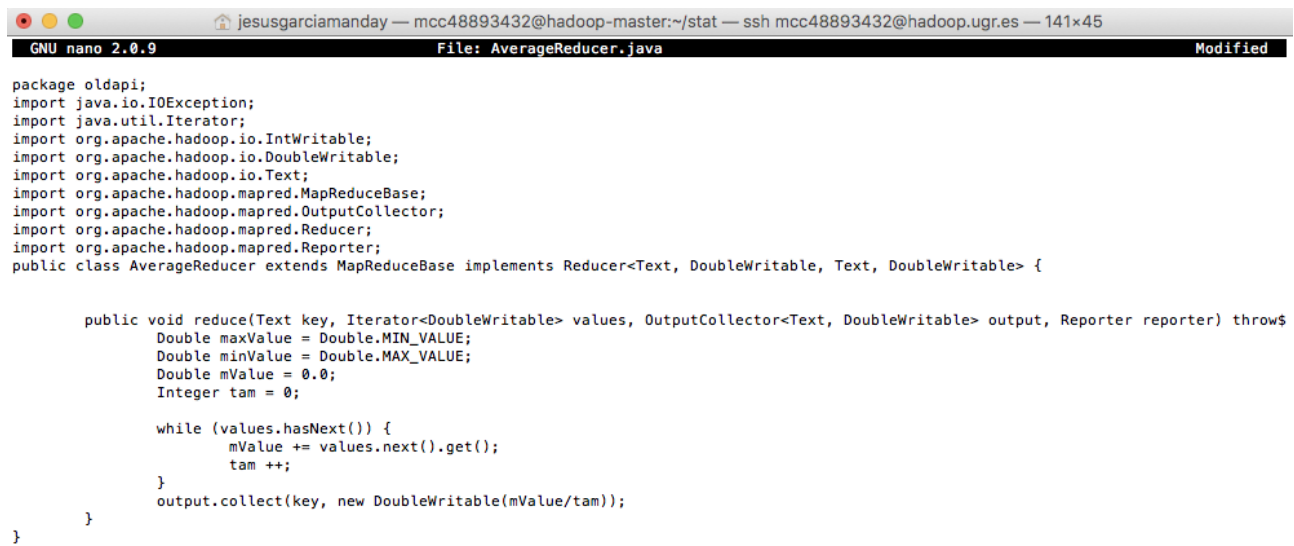
Figura 31: Clase Average.



```
package oldapi;
import java.io.IOException;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.LongWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.Mapper;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reporter;
public class AverageMapper extends MapReduceBase implements Mapper<LongWritable, Text, Text, DoubleWritable> {
    private static final int MISSING = 0;
    public static int col=5;

    public void map(LongWritable key, Text value, OutputCollector<Text, DoubleWritable> output, Reporter reporter) throws IOException {
        String line = value.toString();
        String[] parts = line.split(",");
        output.collect(new Text("1"), new DoubleWritable(Double.parseDouble(parts[col])));
    }
}
```

Figura 32: Clase AverageMapper.



```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 141x45
GNU nano 2.0.9 File: AverageReducer.java Modified

package oldapi;
import java.io.IOException;
import java.util.Iterator;
import org.apache.hadoop.io.IntWritable;
import org.apache.hadoop.io.DoubleWritable;
import org.apache.hadoop.io.Text;
import org.apache.hadoop.mapred.MapReduceBase;
import org.apache.hadoop.mapred.OutputCollector;
import org.apache.hadoop.mapred.Reducer;
import org.apache.hadoop.mapred.Reporter;

public class AverageReducer extends MapReduceBase implements Reducer<Text, DoubleWritable, Text, DoubleWritable> {

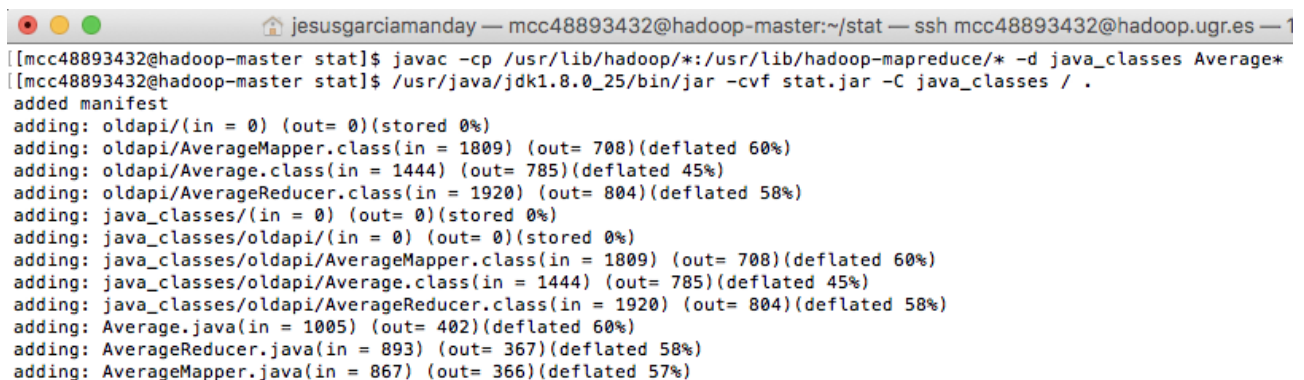
    public void reduce(Text key, Iterator<DoubleWritable> values, OutputCollector<Text, DoubleWritable> output, Reporter reporter) throws $
        Double maxValue = Double.MIN_VALUE;
        Double minValue = Double.MAX_VALUE;
        Double mValue = 0.0;
        Integer tam = 0;

        while (values.hasNext()) {
            mValue += values.next().get();
            tam ++;
        }
        output.collect(key, new DoubleWritable(mValue/tam));
    }
}

```

Figura 33: Clase AverageReducer.

Con las funciones creadas en su correspondiente clase pasamos a compilar y a ejecutar.



```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 1
[mcc48893432@hadoop-master stat]$ javac -cp /usr/lib/hadoop-*/:/usr/lib/hadoop-mapreduce/* -d java_classes Average*
[mcc48893432@hadoop-master stat]$ /usr/java/jdk1.8.0_25/bin/jar -cvf stat.jar -C java_classes / .
added manifest
adding: oldapi/(in = 0) (out= 0)(stored 0%)
adding: oldapi/AverageMapper.class(in = 1809) (out= 708)(deflated 60%)
adding: oldapi/Average.class(in = 1444) (out= 785)(deflated 45%)
adding: oldapi/AverageReducer.class(in = 1920) (out= 804)(deflated 58%)
adding: java_classes/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/(in = 0) (out= 0)(stored 0%)
adding: java_classes/oldapi/AverageMapper.class(in = 1809) (out= 708)(deflated 60%)
adding: java_classes/oldapi/Average.class(in = 1444) (out= 785)(deflated 45%)
adding: java_classes/oldapi/AverageReducer.class(in = 1920) (out= 804)(deflated 58%)
adding: Average.java(in = 1005) (out= 402)(deflated 60%)
adding: AverageReducer.java(in = 893) (out= 367)(deflated 58%)
adding: AverageMapper.java(in = 867) (out= 366)(deflated 57%)

```

Figura 34: Compilamos y ejecutamos (I).

```

jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es — 141x45
[mcc48893432@hadoop-master stat]$ hadoop jar stat.jar oldapi.Average /tmp/BDCC/datasets/ECBDL14/ECBDL14_10tst.data ./stat/output8/
17/05/20 18:21:37 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 18:21:37 INFO client.RMPProxy: Connecting to ResourceManager at hadoop-master/192.168.10.1:8032
17/05/20 18:21:37 WARN mapreduce.JobResourceUploader: Hadoop command-line option parsing not performed. Implement the Tool interface and execute your application with ToolRunner to remedy this.
17/05/20 18:21:37 INFO mapred.FileInputFormat: Total input paths to process : 1
17/05/20 18:21:38 INFO mapreduce.JobSubmitter: number of splits:2
17/05/20 18:21:38 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1494408081774_0290
17/05/20 18:21:38 INFO impl.YarnClientImpl: Submitted application application_1494408081774_0290
17/05/20 18:21:38 INFO mapreduce.Job: The url to track the job: http://hadoop.ugr.es:8088/proxy/application_1494408081774_0290/
17/05/20 18:21:38 INFO mapreduce.Job: Running job: job_1494408081774_0290
17/05/20 18:21:42 INFO mapreduce.Job: Job job_1494408081774_0290 running in uber mode : false
17/05/20 18:21:42 INFO mapreduce.Job: map 0% reduce 0%
17/05/20 18:21:52 INFO mapreduce.Job: map 100% reduce 0%
17/05/20 18:21:56 INFO mapreduce.Job: map 100% reduce 75%
17/05/20 18:21:57 INFO mapreduce.Job: map 100% reduce 81%
17/05/20 18:21:59 INFO mapreduce.Job: map 100% reduce 88%
17/05/20 18:22:00 INFO mapreduce.Job: map 100% reduce 100%
17/05/20 18:22:01 INFO mapreduce.Job: Job job_1494408081774_0290 completed successfully
17/05/20 18:22:01 INFO mapreduce.Job: Counters: 50
  File System Counters
    FILE: Number of bytes read=2168431
    FILE: Number of bytes written=6521412
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=102749934
    HDFS: Number of bytes written=21
    HDFS: Number of read operations=54
    HDFS: Number of large read operations=0
    HDFS: Number of write operations=32
  Job Counters
    Launched map tasks=2
    Launched reduce tasks=16
    Data-local map tasks=1
    Rack-local map tasks=1
    Total time spent by all maps in occupied slots (ms)=108864
    Total time spent by all reduces in occupied slots (ms)=1851759
    Total time spent by all map tasks (ms)=15552
    Total time spent by all reduce tasks (ms)=37791
    Total vcore-seconds taken by all map tasks=15552
    Total vcore-seconds taken by all reduce tasks=37791
    Total megabyte-seconds taken by all map tasks=108864000
    Total megabyte-seconds taken by all reduce tasks=1889550000

```

Figura 35: Compilamos y ejecutamos (I).

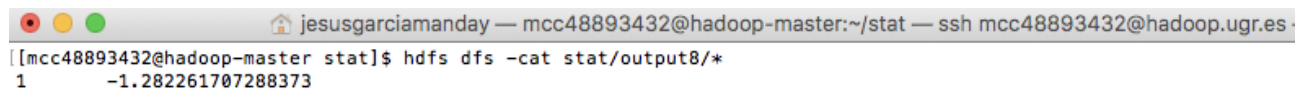
```

Map-Reduce Framework
  Map input records=2897917
  Map output records=2897917
  Map output bytes=28979170
  Map output materialized bytes=2168457
  Input split bytes=234
  Combine input records=0
  Combine output records=0
  Reduce input groups=1
  Reduce shuffle bytes=2168457
  Reduce input records=2897917
  Reduce output records=1
  Spilled Records=5795834
  Shuffled Maps =32
  Failed Shuffles=0
  Merged Map outputs=32
  GC time elapsed (ms)=337
  CPU time spent (ms)=35700
  Physical memory (bytes) snapshot=7900184576
  Virtual memory (bytes) snapshot=984125603840
  Total committed heap usage (bytes)=19167969280
Shuffle Errors
  BAD_ID=0
  CONNECTION=0
  IO_ERROR=0
  WRONG_LENGTH=0
  WRONG_MAP=0
  WRONG_REDUCE=0
File Input Format Counters
  Bytes Read=102749700
File Output Format Counters
  Bytes Written=21

```

Figura 36: Compilamos y ejecutamos (III).

Y por último comprobamos el resultado.



```
jesusgarciamanday — mcc48893432@hadoop-master:~/stat — ssh mcc48893432@hadoop.ugr.es  
[[mcc48893432@hadoop-master stat]$ hdfs dfs -cat stat/output8/*  
1 -1.282261707288373
```

Figura 37: Comprobando resultado.

8. Obtener la media de todas las variables (salvo la clase).
9. Comprobar si el conjunto de datos ECBDL es balanceado o no balanceado, es decir, que el ratio entre clases sea menor o mayor que 1.5 respectivamente.
10. Cálculo del coeficiente de correlación entre todas las parejas de variables.