

# Big Data Analytics

## Lab 01 - Hadoop MapReduce in the cloud

Damien Rochat, Dorian Magnin, and Nelson Rocha  
Master of Science in Engineering  
University of Applied Sciences and Arts Western Switzerland

March 8, 2019

### 1 Task 2: Using Elastic MapReduce

#### 1.1 EMR console summary


Cluster: Max. Temperature   Terminated   Steps completed

Summary   Monitoring   Hardware   Events   Steps   Bootstrap actions

**Connections:** --

**Master public DNS:** ec2-34-204-89-126.compute-1.amazonaws.com [SSH](#)

**Tags:** --

Summary	Configuration details
<b>ID:</b> j-3TRYUL5TQBRBB	<b>AMI version:</b> 2.4.11
<b>Creation date:</b> 2019-03-02 10:04 (UTC+1)	<b>Hadoop distribution:</b> Amazon 1.0.3
<b>End date:</b> 2019-03-02 10:25 (UTC+1)	<b>Applications:</b> --
<b>Elapsed time:</b> 21 minutes	<b>Log URI:</b> s3://aws-logs-680384311199-us-east-1/elasticmapreduce/ 
<b>Auto-terminate:</b> Yes	<b>EMRFS consistent view:</b> Disabled
<b>Termination protection:</b> Off	




Network and hardware	Security and access
<b>Availability zone:</b> us-east-1a	<b>Key name:</b> damien_yoda
<b>Subnet ID:</b> <a href="#">subnet-2f850773</a> 	<b>EC2 instance profile:</b> EMR_EC2_DefaultRole
<b>Master:</b> Terminated 1 m1.small	<b>EMR role:</b> EMR_DefaultRole
<b>Core:</b> Terminated 2 m1.small	<b>Visible to all users:</b> All <a href="#">Change</a>
<b>Task:</b> --	<b>Security groups for</b> <a href="#">sg-07c8d4be0aaf8abd6</a> 
	<b>Master:</b> (ElasticMapReduce-master)
	<b>Security groups for</b> <a href="#">sg-04bf7c70a00696c58</a> 
	<b>Core &amp; Task:</b> (ElasticMapReduce-slave)

Figure 1: Elastic MapReduce summary

## 1.2 Max temperature chart

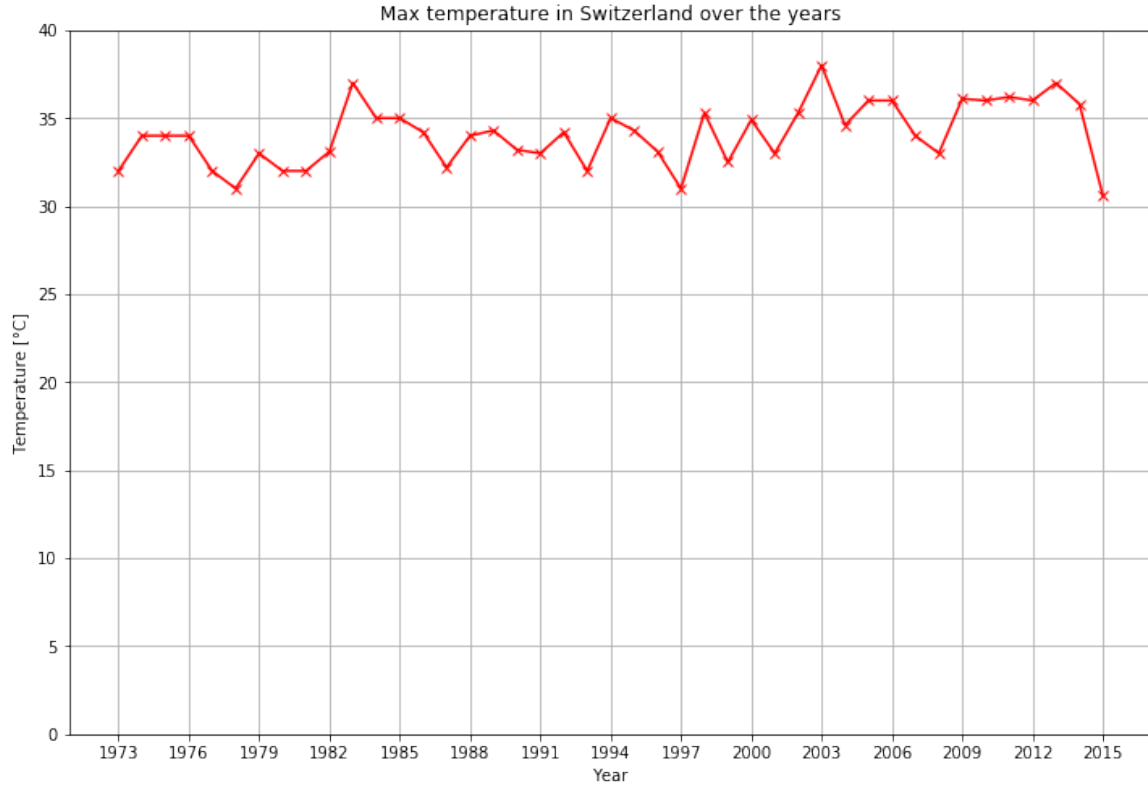


Figure 2: Max temperature chart

The overall highest recorded temperature is of 38.0 °C in 2003, year of the European heat wave.

## 1.3 EC2 instances

We used 3 m1.small instances (1 master and 2 cores).

For each one, the price of the on-demand EC2 is \$0.044 + \$0.011 for the EMR per hour. Our cluster ran during 21 minutes.

The total cost is of \$0.05775  $((\$0.044 + \$0.011) * 3 * 0.35 = \$0.05775)$ .

## 1.4 EMR job

From the job log file, the Mappers processed **2'831'380** input records (MAP\_INPUT\_RECORDS) and produced **2'821'078** records (MAP\_OUTPUT\_RECORDS).

The Reducers processed **2'821'078** input records (REDUCE\_INPUT\_RECORDS) and produced **43** records (REDUCE\_OUTPUT\_RECORDS).

## 2 Task 3: Writing a MapReduce program

### 2.1 Source code

#### 2.1.1 Mapper

```
1  #!/usr/bin/env python
2  #
3  # max_temperature_map.py - Count temperature from NCDC Global
4  #                           Hourly Data - Mapper part
5
6  import re    # import regular expressions
7  import sys   # import system-specific parameters and functions
8
9  map_count = {}
10
11 # loop through the input, line by line
12 for line in sys.stdin:
13     # remove leading and trailing whitespace
14     val = line.strip()
15     # extract values for temperature and quality indicator
16     temp = val[87:92]
17     q = val[92:93]
18     # temperature is valid if not +9999 and quality indicator is
19     # one of 0, 1, 4, 5 or 9
20     if (temp != "+9999" and re.match("[01459]", q)):
21
22         # binning of the temperature
23         temp = int(float(temp)/10)
24
25         map_count[temp] = map_count.get(temp, 0) + 1
26
27 # produce output
28 for temp, count in map_count.iteritems():
29     print "%s\t%s" % (temp, count)
```

Listing 1: max\_temperature\_map.py source code

#### 2.1.2 Reducer

```
1  #!/usr/bin/env python
2  #
3  # max_temperature_reduce.py - Count temperature from NCDC Global
4  #                           Hourly Data - Reducer part
5
6  import sys
7
8  last_key = None
9  count = 0
10 # loop through the input, line by line
11 for line in sys.stdin:
12     # each line contains a key and a value separated by a tab character
13     (key, val) = line.strip().split("\t")
14     # Hadoop has sorted the input by key, so we get the values
15     # for the same key immediately one after the other.
16     # Test if we just got a new key, in this case output the count
```

```

17 # temperature for the previous key and reinitialize the variables.
18 # If not, keep counting temperature.
19 if last_key and last_key != key:
20     print "%s\t%s" % (last_key, count)
21     count = int(val)
22 else:
23     count = count+int(val)
24     last_key = key
25
26 # we've reached the end of the file, output what is left
27 if last_key:
28     print "%s\t%s" % (last_key, count)

```

Listing 2: max\_temperature\_reduce.py source code

We also tried a non-optimized version of the code (without "In-Mapper combining") and obtained a time difference of 3 minutes (10 minutes instead of 7 minutes) on the EMR step duration.

## 2.2 Stats

How often does the temperature 22 degrees celsius occur? **56'530 times**

What is the lowest and highest temperature occurring? **min: -25 °C, max: 38 °C**

Which temperature occurs most often? **13 °C with 114'613 occurrences**

## 2.3 Histogram

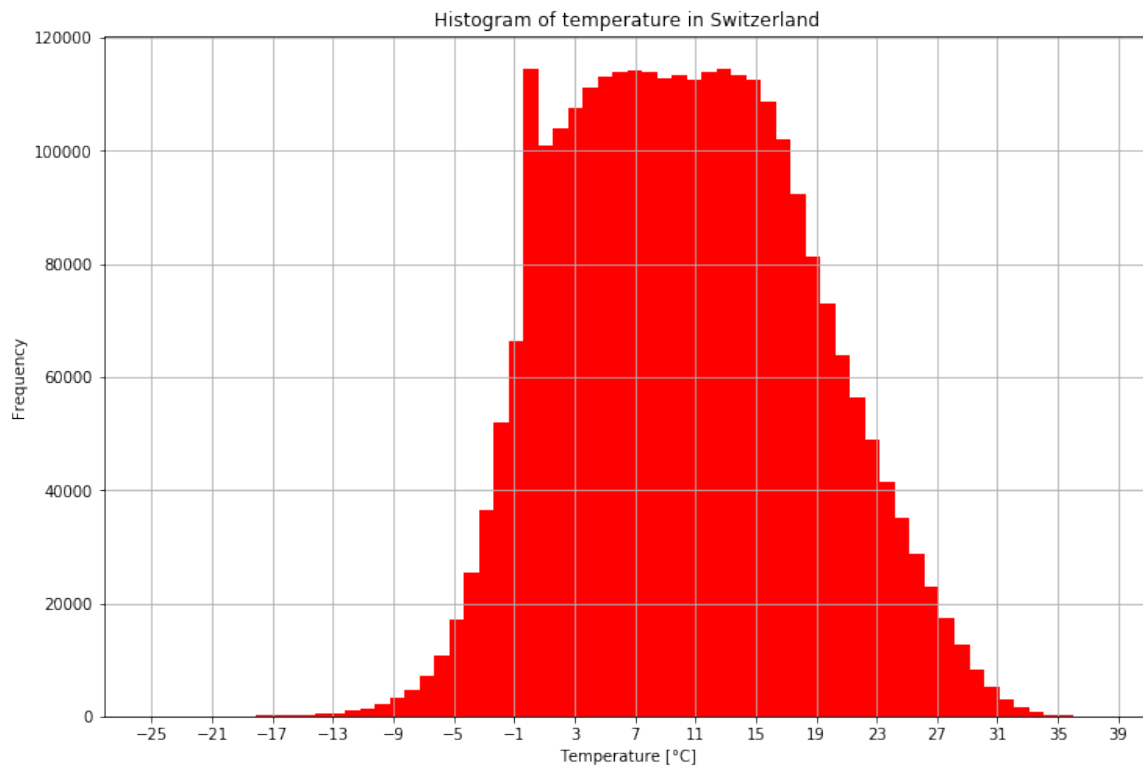


Figure 3: Histogram of temperature in Switzerland