Text Mining and Machine Learning with Apache Spark

POLTEXT 2019, tutorial D

Zoltan Kacsuk, Centre for Social Sciences (HU) & IAAI, Stuttgart Media University

Our plan for today

1) Understanding the role of Apache Spark in the big data ecosystem
2) Apache Spark and Hadoop architectures in a nutshell
3) Configuring and using the Spark context
4) Using a Hadoop Distributed File System with the cluster
5) Operating the cluster via an RStudio Server and sparklyr
6) The differences in available functionality of the MLlib APIs
7) How the Hungarian CAP Project in enabled by a Spark cluster

Understanding the role of Apache Spark in the big data ecosystem

Big Data

The theoretical approach

a) Volume
b) Variety
c) Velocity

Big Data

The theoretical approach

a) Volume
b) Variety
c) Velocity

The practical approach

a) Doesn’t fit
b) Too slow

It can feel a bit overwhelming. (And this is just the tip of the iceberg.)

Implementing a machine learning process: the **where**

Supercomputer: One very powerful machine

Cluster: Lots of machines working together to create a powerful entity

Cloud: Infrastructure as a service

Containers: easier deployment of “virtual machines”

Container and cluster orchestration tools: easier deployment of clusters

Implementing a machine learning process: the **where**

Supercomputer: One very powerful machine

Cluster: Lots of machines working together to create a powerful entity

openstack.

Cloud: Infrastructure as a service

Containers: easier deployment of “virtual machines”

Container and cluster orchestration tools: easier deployment of clusters

Implementing a machine learning process: the **where**

Supercomputer: One very powerful machine

Cluster: Lots of machines working together to create a powerful entity

Cloud: Infrastructure as a service

Containers: easier deployment of “virtual machines”

Container and cluster orchestration tools: easier deployment of clusters

Implementing a machine learning process: the **where**

Supercomputer: One very powerful machine

Cluster: Lots of machines working together to create a powerful entity

openstack.

Cloud: Infrastructure as a service

Containers: easier deployment of “virtual machines”

docker

Container and cluster orchestration tools: easier deployment of clusters

Implementing a machine learning process: the **where**

Supercomputer: One very powerful machine

Cluster: Lots of machines working together to create a powerful entity

openstack

Cloud: Infrastructure as a service

Containers: easier deployment of “virtual machines”

docker

Container and cluster orchestration tools: easier deployment of clusters

Apache MESOS

Cloud service providers with complete vertical coverage

- Infrastructure as a service
- Platform as a service
- Software as a service

E.g.: Amazon Web Services, Google Cloud Platform, Microsoft Azure

Implementing a machine learning process: the **how**

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations is achieved

Data distribution and **I/O operations**

Implementing a machine learning process: the **how**

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations is achieved

Data distribution and **I/O operations**

Implementing a machine learning process: the **how**

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations is achieved

Data distribution and **I/O operations**

Implementing a machine learning process: the **how**

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations is achieved

Data distribution and **I/O operations**

Implementing a machine learning process: the **how**

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations

Data distribution and **I/O operations**

Implementing a machine learning process: the **how**

The type of **statistical model** MAHOUT

The **exact formula** being implemented MAHOUT

The way the **algorithm** works MAHOUT

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved APACHE Spark

The way the **control of the distribution** of the operations Apache MESOS APACHE Spark

Data distribution and **I/O operations** HIVE

APACHE hadoop

Implementing a machine learning process: the **how**

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations

Data distribution and **I/O operations**

Implementing a machine learning process: the **how**

The type of **statistical model**

The **exact formula** being implemented

This is the Spark Machine Learning Library

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations

Data distribution and **I/O operations**

Implementing a machine learning process: the **how**

The type of **statistical model** MAHOUT H$_2$O.ai Apache Spark

The **exact formula** being implemented MAHOUT H$_2$O.ai Apache Spark

---

The way the **algorithm** works MAHOUT H$_2$O.ai Apache Spark

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved Apache Spark

The way the **control of the distribution** of the operations Apache MESOS Apache Spark

Data distribution and **I/O operations** HIVE

Apache hadoop

Implementing a machine learning process: the **how**

The type of **statistical model** MAHOUT H₂O.ai APACHE Spark

The **exact formula** being implemented MAHOUT H₂O.ai APACHE Spark

---

The way the **algorithm** works MAHOUT H₂O.ai APACHE Spark

The way **individual calculations** are computed MAHOUT H₂O.ai APACHE ARROW

The way the **distribution of the operations** is achieved APACHE Spark

The way the **control of the distribution** of the operations Apache MESOS APACHE Spark

Data distribution and **I/O operations** HIVE

APACHE hadoop

Implementing a machine learning process: the **how**

The type of **statistical model** MAHOUT H_2O.ai APACHE Spark

The **exact formula** being implemented MAHOUT H_2O.ai APACHE Spark

---

The way the **algorithm** works MAHOUT H_2O.ai APACHE Spark

The way **individual calculations** are computed MAHOUT H_2O.ai APACHE ARROW

The way the **distribution of the operations** is achieved APACHE Spark

The way the **control of the distribution** of the operations Apache MESOS APACHE Spark

Data distribution and **I/O operations** APACHE hadoop HIVE

The sediments of evolving big data technology

Data distribution: Apache Hadoop HDFS

Distribution of operations: Apache Spark

Rest of the stack: pick & combine

The sediments of evolving big data technology

Data distribution: Apache Hadoop HDFS

Distribution of operations: Apache Spark

Rest of the stack: pick & combine

APACHE hadoop + APACHE Spark

The sediments of evolving big data technology

Data distribution: Apache Hadoop HDFS

Distribution of operations: Apache Spark

Rest of the stack: pick & combine

APACHE hadoop + APACHE Spark

APACHE hadoop + APACHE Spark + MAHOUT

The sediments of evolving big data technology

Data distribution: Apache Hadoop HDFS

Distribution of operations: Apache Spark

Rest of the stack: pick & combine

APACHE hadoop + APACHE Spark

APACHE hadoop + APACHE Spark + MAHOUT

APACHE hadoop + APACHE Spark + APACHE ARROW

The sediments of evolving big data technology

Data distribution: Apache Hadoop HDFS

Distribution of operations: Apache Spark

Rest of the stack: pick & combine

APACHE hadoop + APACHE Spark

APACHE hadoop + APACHE Spark + MAHOUT

APACHE hadoop + APACHE Spark + APACHE ARROW

APACHE hadoop + HIVE + APACHE Spark

The sediments of evolving big data technology

Data distribution: Apache Hadoop HDFS

Distribution of operations: Apache Spark

Rest of the stack: pick & combine

APACHE hadoop + APACHE Spark

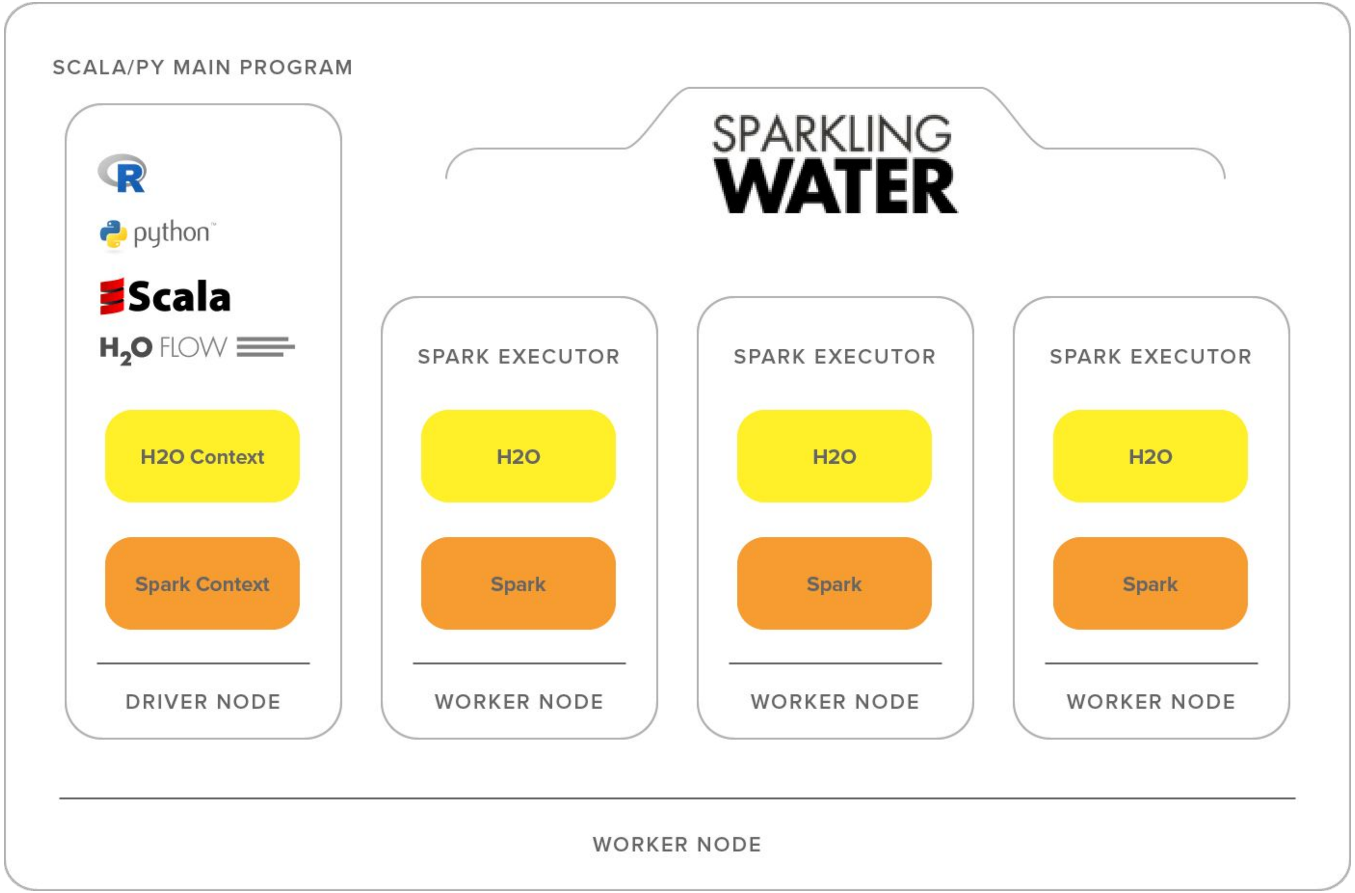
APACHE hadoop + APACHE Spark + MAHOUT

APACHE hadoop + APACHE Spark + APACHE ARROW

APACHE hadoop + HIVE + APACHE Spark

APACHE hadoop + Apache MESOS + APACHE Spark + MAHOUT

APACHE Spark? ... !



SCALA/PY MAIN PROGRAM

SPARKLING WATER

R
python
Scala
H_2O FLOW

H2O Context

Spark Context

DRIVER NODE

SPARK EXECUTOR

H2O

Spark

WORKER NODE

SPARK EXECUTOR

H2O

Spark

WORKER NODE

SPARK EXECUTOR

H2O

Spark

WORKER NODE

WORKER NODE

APACHE hadoop

+

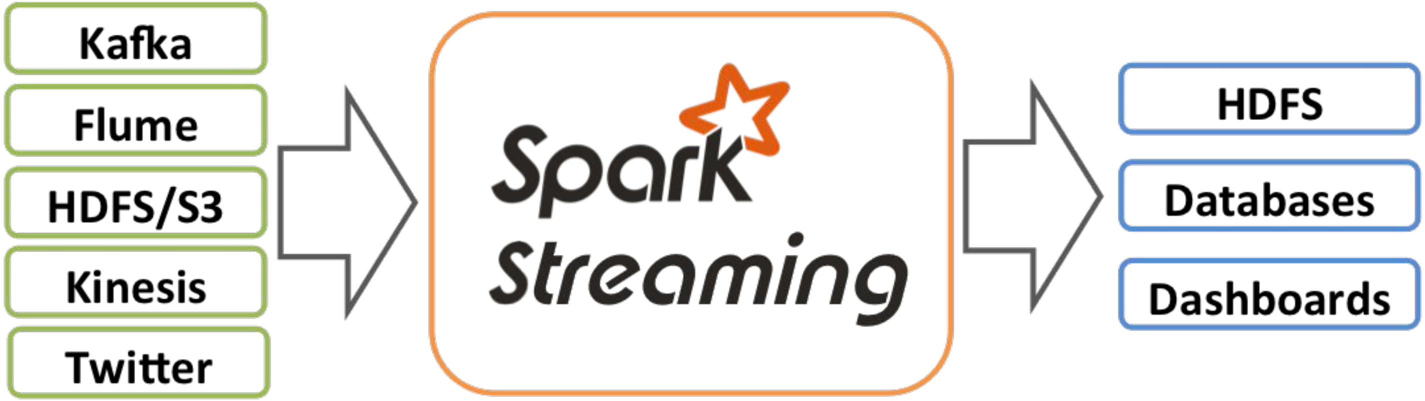
APACHE Spark

+

H_2O.ai

Image source and details at: https://www.h2o.ai/products/h2o-sparkling-water/

Spark also handles **streaming data** and **graph data**



Kafka

Flume

HDFS/S3

Kinesis

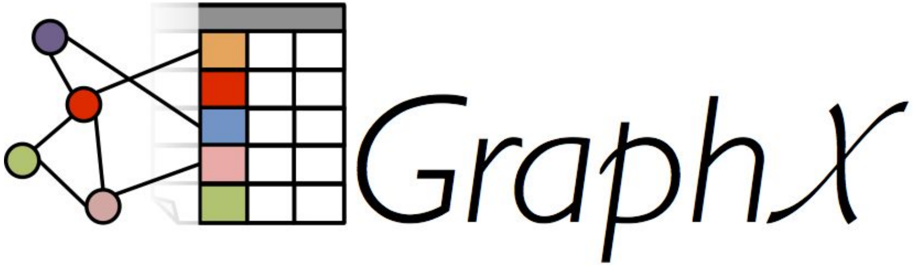
Twitter

Spark Streaming

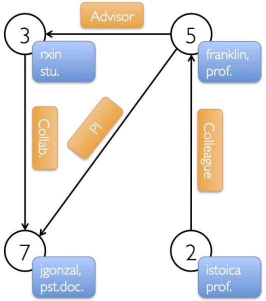
HDFS

Databases

Dashboards



Property Graph



Vertex Table

Id	Property (V)
3	(rxin, student)
7	(jgonzal, postdoc)
5	(franklin, professor)
2	(istoica, professor)

Edge Table

SrcId	DstId	Property (E)
3	7	Collaborator
5	3	Advisor
2	5	Colleague
5	7	PI

Images from and more details at: https://spark.apache.org

Apache Spark and Hadoop architectures in a nutshell

Spark architecture basics

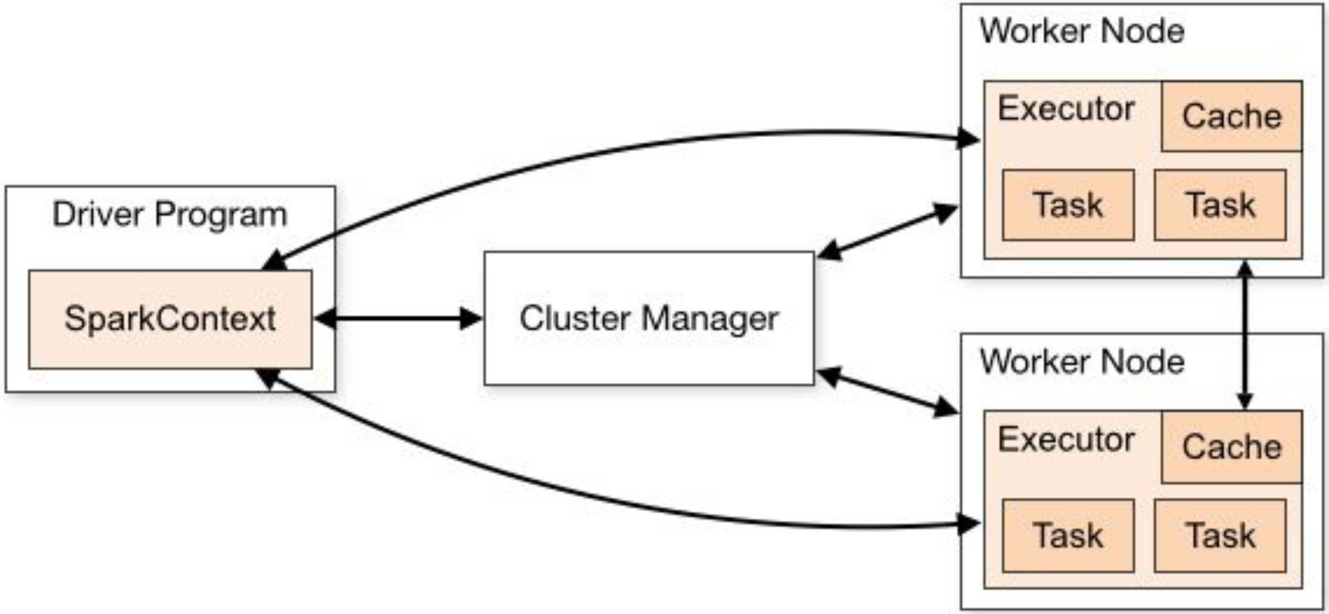


Image source and further details: https://spark.apache.org/docs/latest/cluster-overview.html

Spark architecture basics

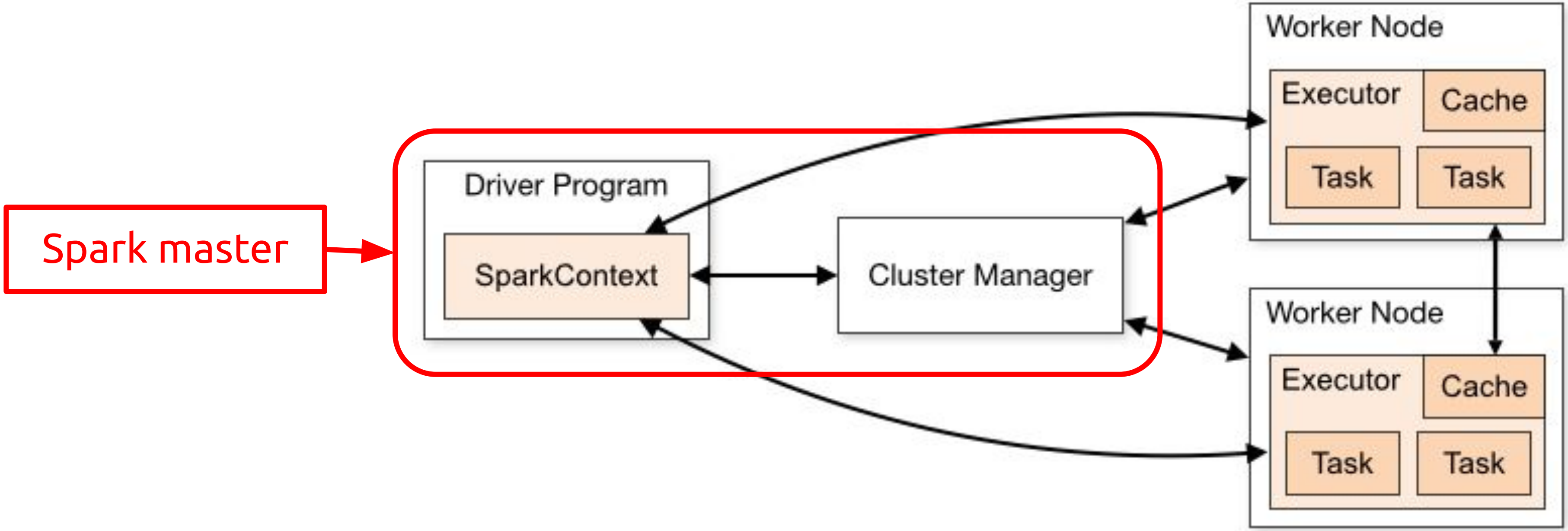


Image source and further details: https://spark.apache.org/docs/latest/cluster-overview.html

Spark architecture basics

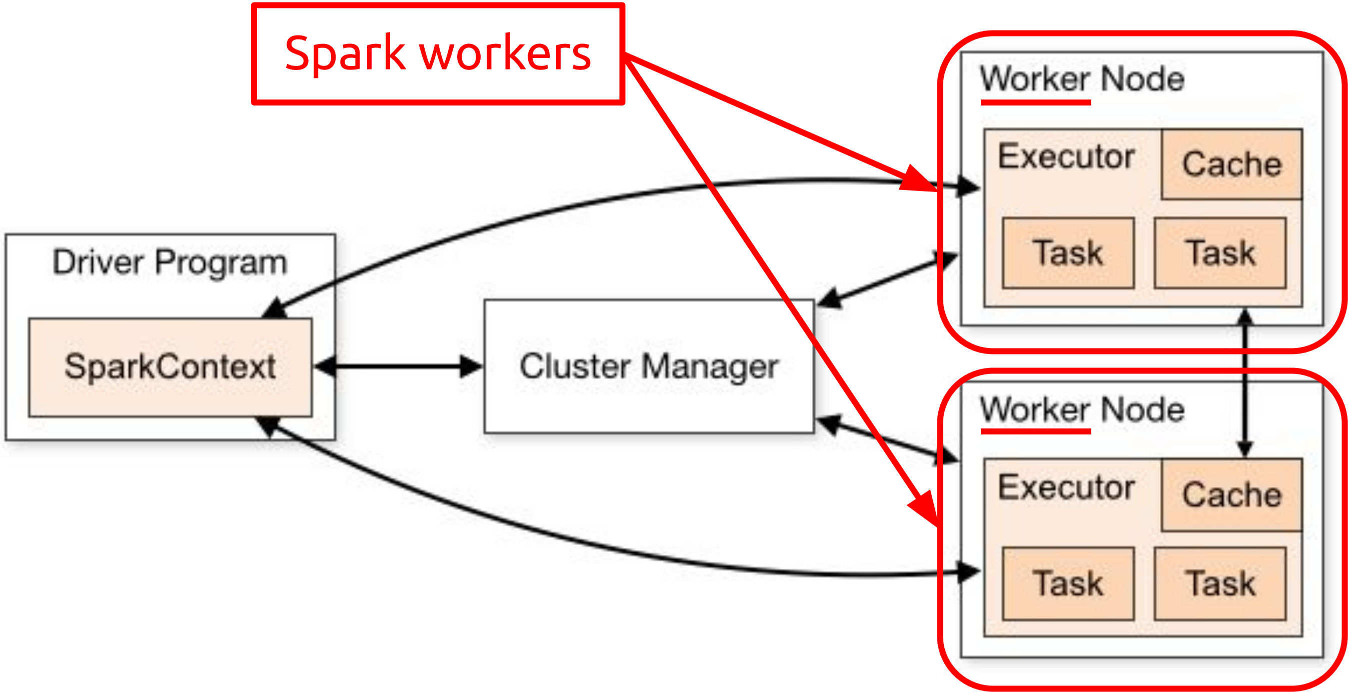
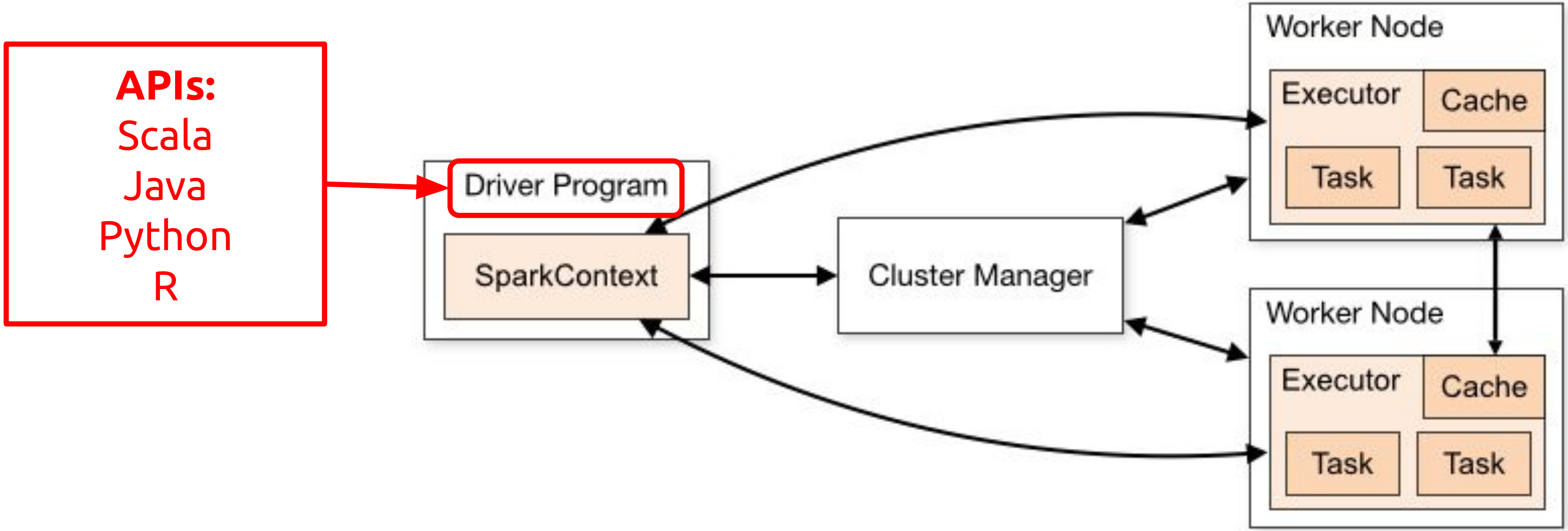


Image source and further details: https://spark.apache.org/docs/latest/cluster-overview.html

Spark architecture basics



APIs:
Scala
Java
Python
R

Image source and further details: https://spark.apache.org/docs/latest/cluster-overview.html

Spark architecture basics

Think of this as a single shell, session or job

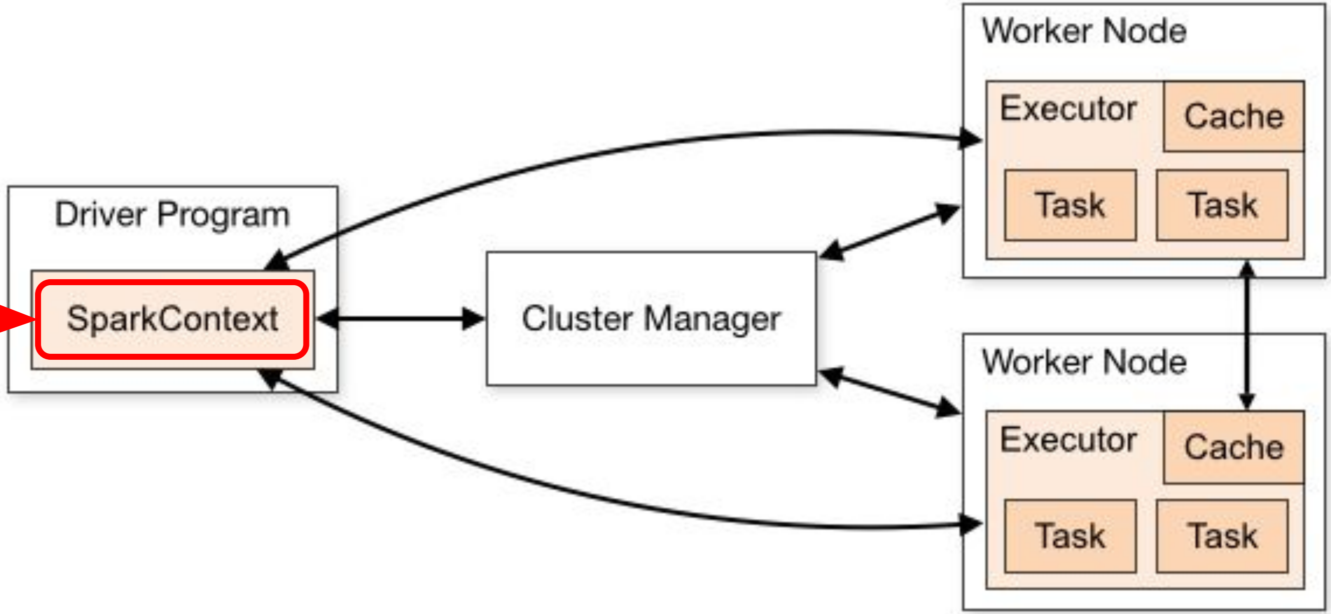
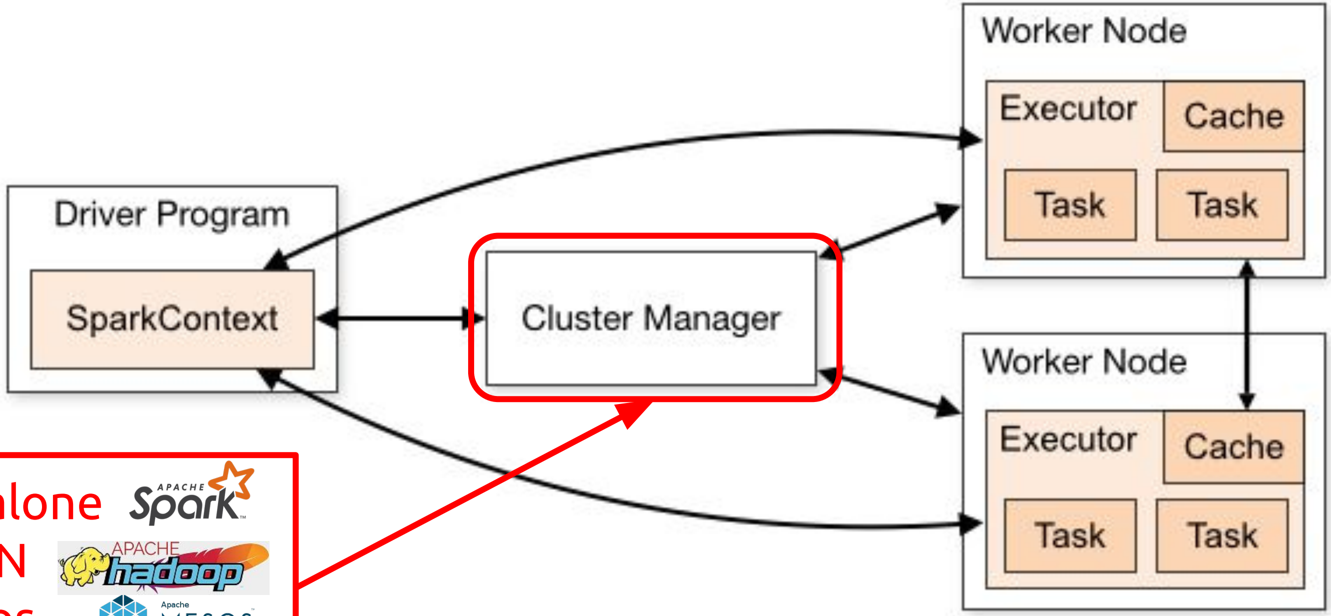


Image source and further details: https://spark.apache.org/docs/latest/cluster-overview.html

Spark architecture basics

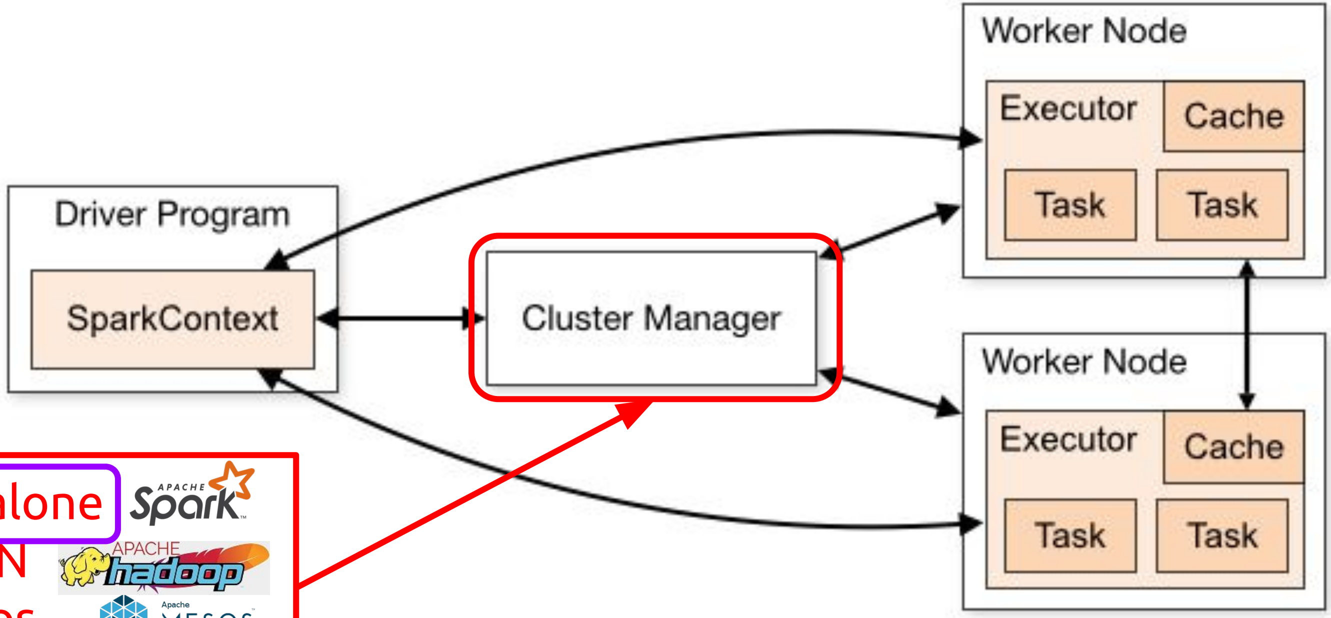


- Spark Standalone
- Hadoop YARN
- Apache Mesos
- Kubernetes

Image source and further details: https://spark.apache.org/docs/latest/cluster-overview.html

Spark architecture basics

IMPORTANT:
Standalone does not equal **local** mode!



Spark Standalone
Hadoop YARN
Apache Mesos
Kubernetes

Image source and further details: https://spark.apache.org/docs/latest/cluster-overview.html

Understanding Hadoop for Spark

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved: **Hadoop MapReduce**

The way the **control of the distribution** of the operations: **Hadoop YARN**

Data distribution and **I/O operations**: **Hadoop Distributed File System (HDFS)**

Understanding Hadoop for Spark

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved: **Hadoop MapReduce**

The way the **control of the distribution** of the operations: **Hadoop YARN**

Data distribution and **I/O operations**: **Hadoop Distributed File System (HDFS)**

The Hadoop Distributed File System (HDFS)

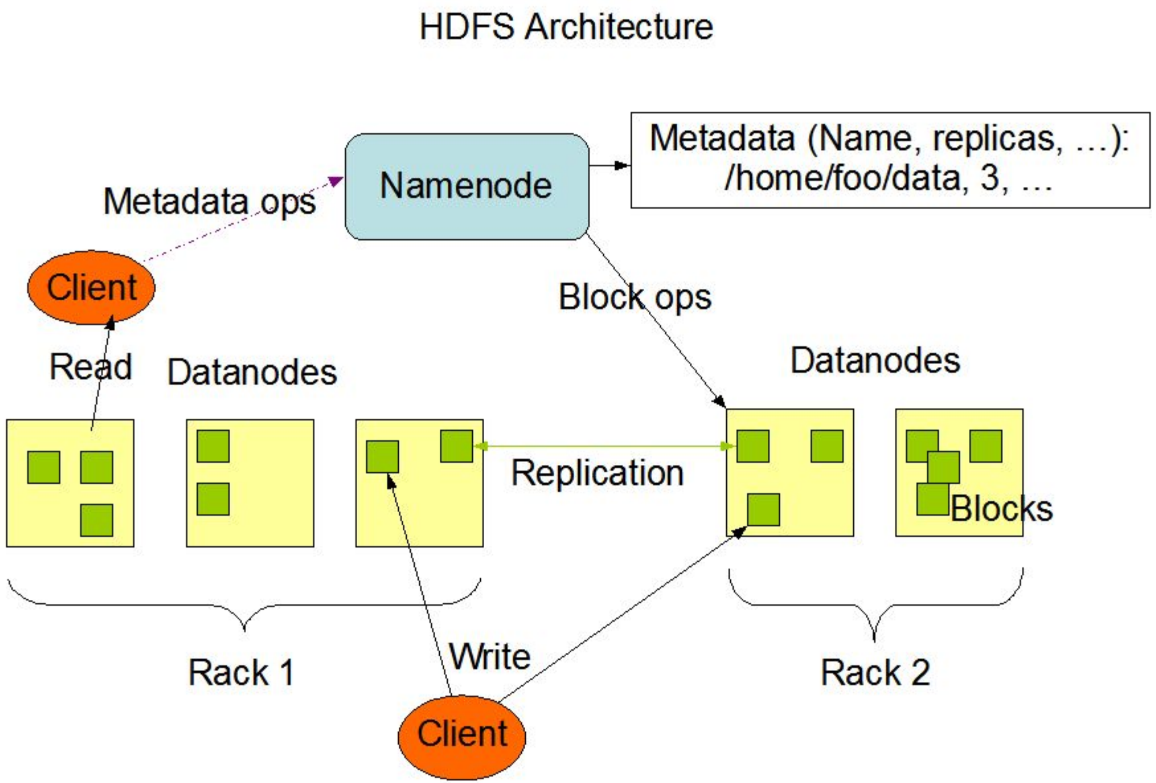
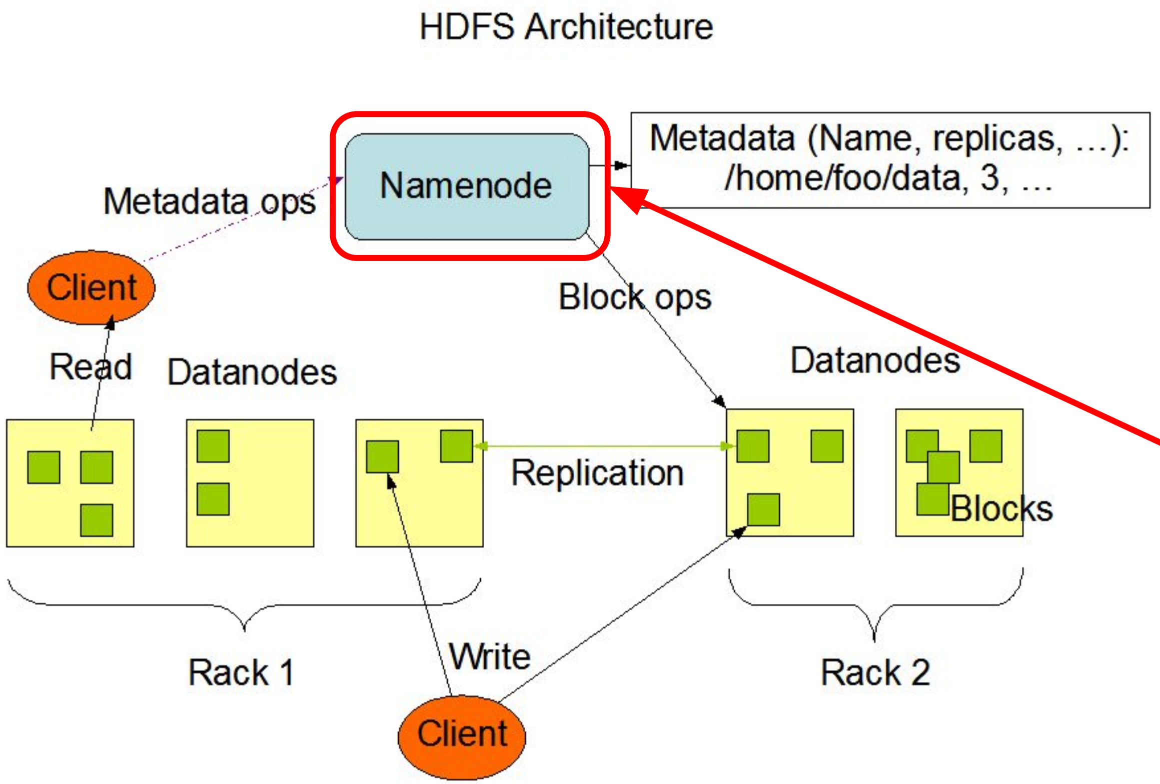


Image source and further details: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

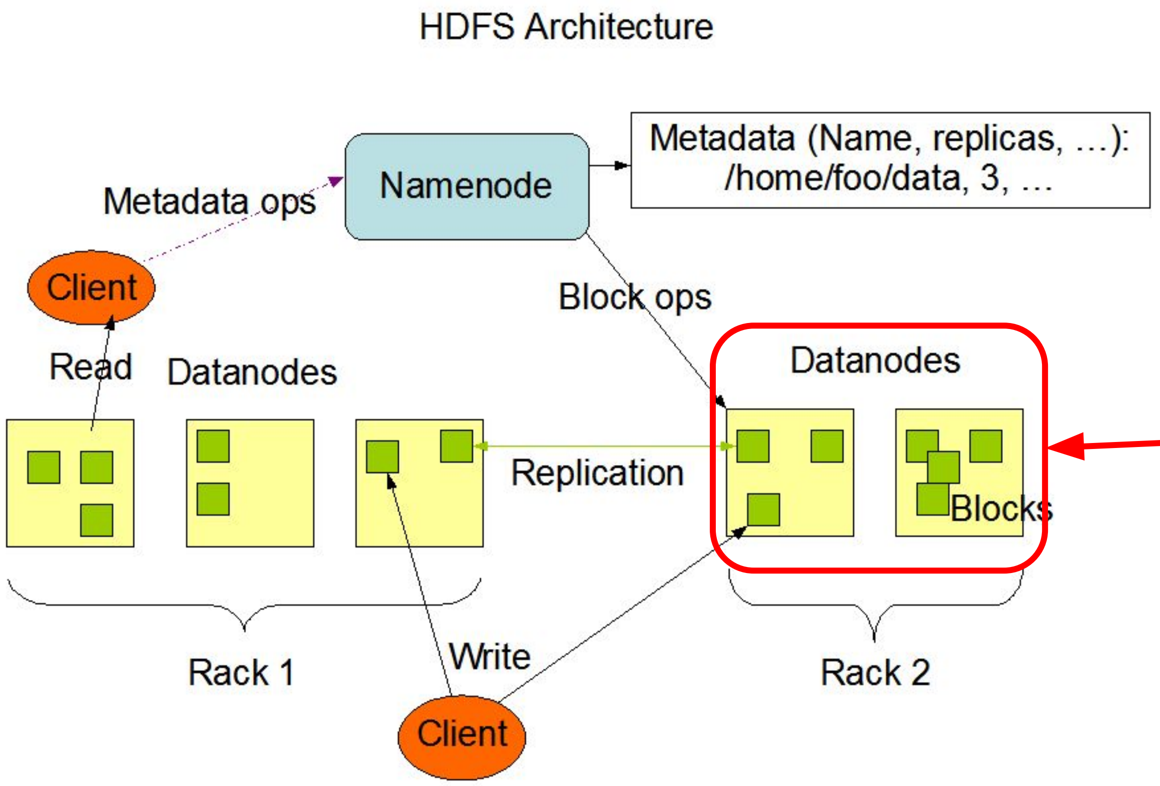
The HDFS NameNode



- This is running **on the Spark master**.
- This is what we see when we browse the filesystem.

Image source and further details: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

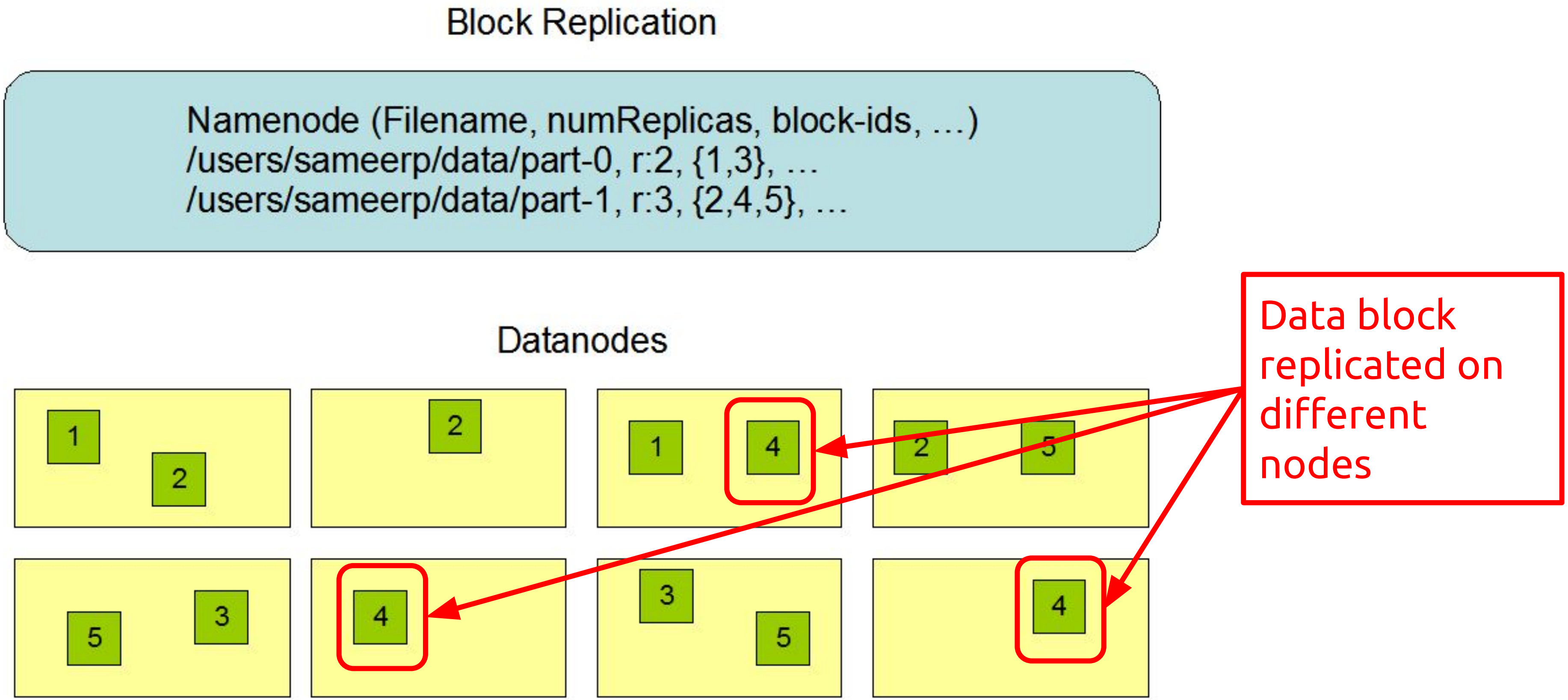
The HDFS DataNodes



- These are running **on the Spark workers**.
- This is where the data actually is.

Image source and further details: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

Data replication and fault tolerance in the HDFS



Block Replication

Namenode (Filename, numReplicas, block-ids, …)
/users/sameerp/data/part-0, r:2, {1,3}, …
/users/sameerp/data/part-1, r:3, {2,4,5}, …

Datanodes

Data block replicated on different nodes

Image source and further details: https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html

Let’s get practical!

The tutorial will discuss

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations

Configuring the Apache Spark cluster

Data distribution and **I/O operations**

The tutorial will discuss

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations

Data distribution and **I/O operations**

Using a Hadoop Distributed File System with the cluster

The tutorial will discuss

The type of **statistical model**

The **exact formula** being implemented

The way the **algorithm** works

Operating the cluster via an RStudio Server and sparklyr

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations

Data distribution and **I/O operations**

The tutorial will discuss

The type of **statistical model**

The differences in available functionality of the MLlib APIs

The **exact formula** being implemented

The way the **algorithm** works

The way **individual calculations** are computed

The way the **distribution of the operations** is achieved

The way the **control of the distribution** of the operations

Data distribution and **I/O operations**

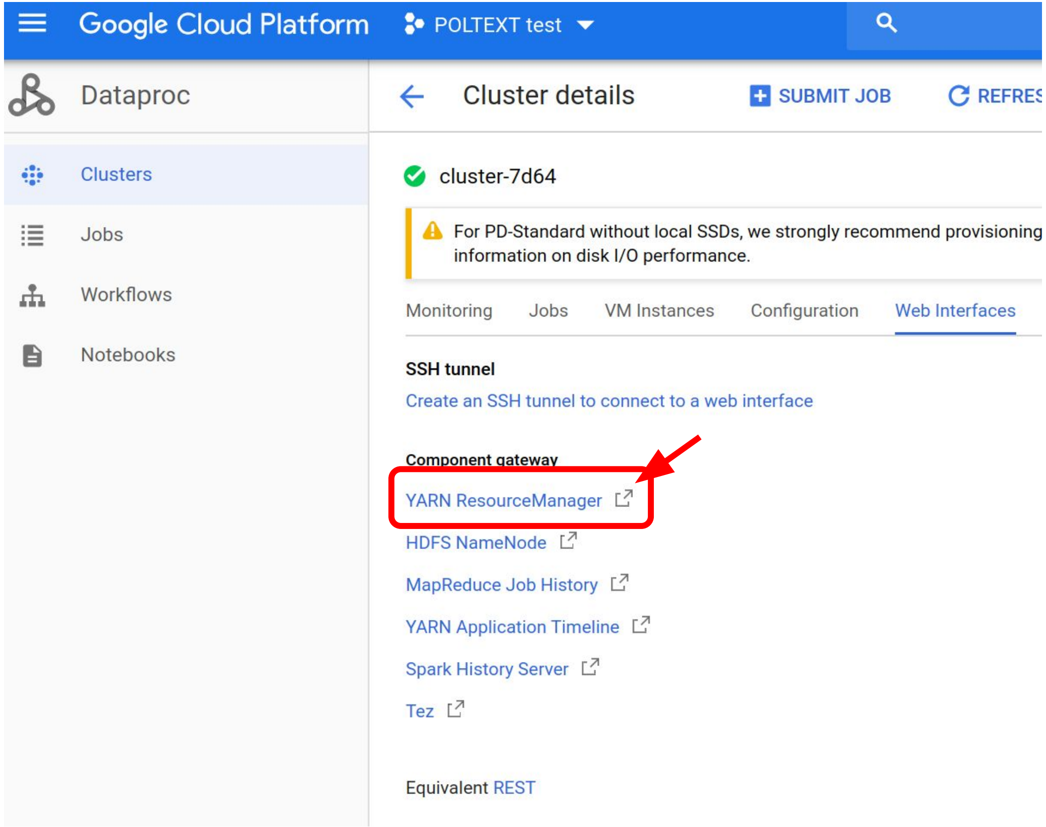
Let’s create a Spark cluster with an RStudio Server set up in the Google Cloud

Steps to create the Spark cluster in Google Cloud with RStudio Server

Follow the steps here:
https://cloud.google.com/solutions/running-rstudio-server-on-a-cloud-dataproc-cluster

Or check out the illustrated guide here:
https://github.com/zkpti/poltext2019-sparktutorial/blob/master/cluster_setup/Setting_up_Google_Cloud_Spark_cluster.pdf

Important:
The cluster manager is set to be **Hadoop YARN** by default in the Google Cloud Dataproc clusters



cluster-7d64

For PD-Standard without local SSDs, we strongly recommend provisioning
information on disk I/O performance.

SSH tunnel

Create an SSH tunnel to connect to a web interface

Component gateway

YARN ResourceManager

HDFS NameNode

MapReduce Job History

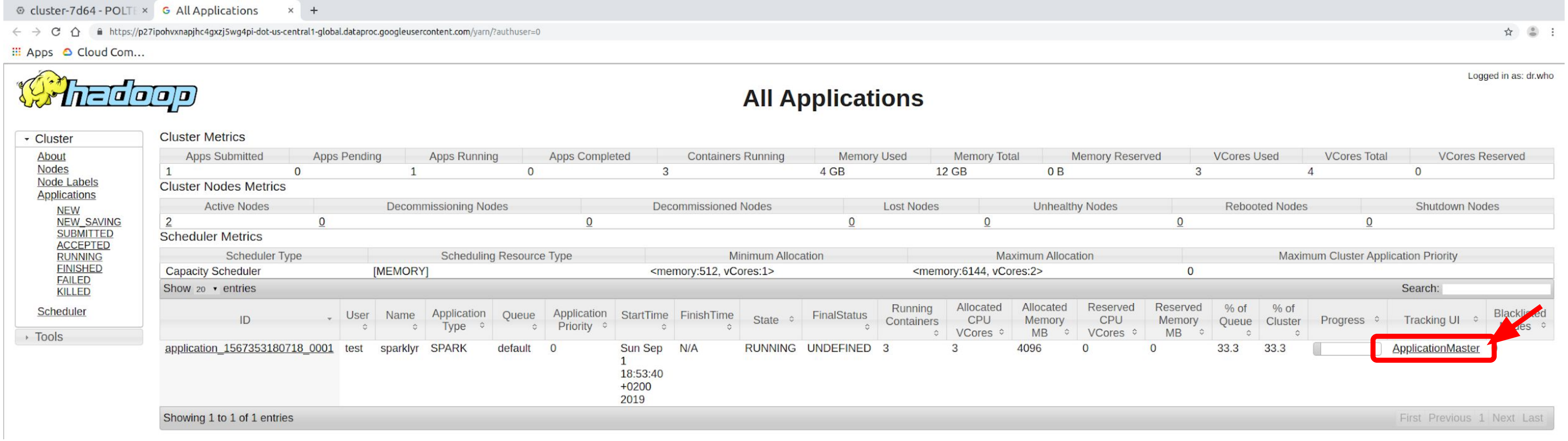
YARN Application Timeline

Spark History Server

Tez

Equivalent REST

We can inspect our YARN cluster manager from our browser. Go to the cluster management interface Web Interfaces tab and press YARN ResourceManager.



cluster-7d64 - POLTE ×
All Applications ×
+
https://p27ipohvxnapjhc4gxzj5wg4pi-dot-us-central1-global.dataproc.googleusercontent.com/yarn/?authuser=0
Apps
Cloud Com...
hadoop

Logged in as: dr.who

All Applications

- Cluster
  - About
  - Nodes
  - Node Labels
  - Applications
    - NEW
    - NEW_SAVING
    - SUBMITTED
    - ACCEPTED
    - RUNNING
    - FINISHED
    - FAILED
    - KILLED
  - Scheduler
- Tools

Cluster Metrics

Apps Submitted	Apps Pending	Apps Running	Apps Completed	Containers Running	Memory Used	Memory Total	Memory Reserved	VCores Used	VCores Total	VCores Reserved
1	0	1	0	3	4 GB	12 GB	0 B	3	4	0

Cluster Nodes Metrics

Active Nodes	Decommissioning Nodes	Decommissioned Nodes	Lost Nodes	Unhealthy Nodes	Rebooted Nodes	Shutdown Nodes
2	0	0	0	0	0	0

Scheduler Metrics

Scheduler Type	Scheduling Resource Type	Minimum Allocation	Maximum Allocation	Maximum Cluster Application Priority
Capacity Scheduler	[MEMORY]	<memory:512, vCores:1>	<memory:6144, vCores:2>	0

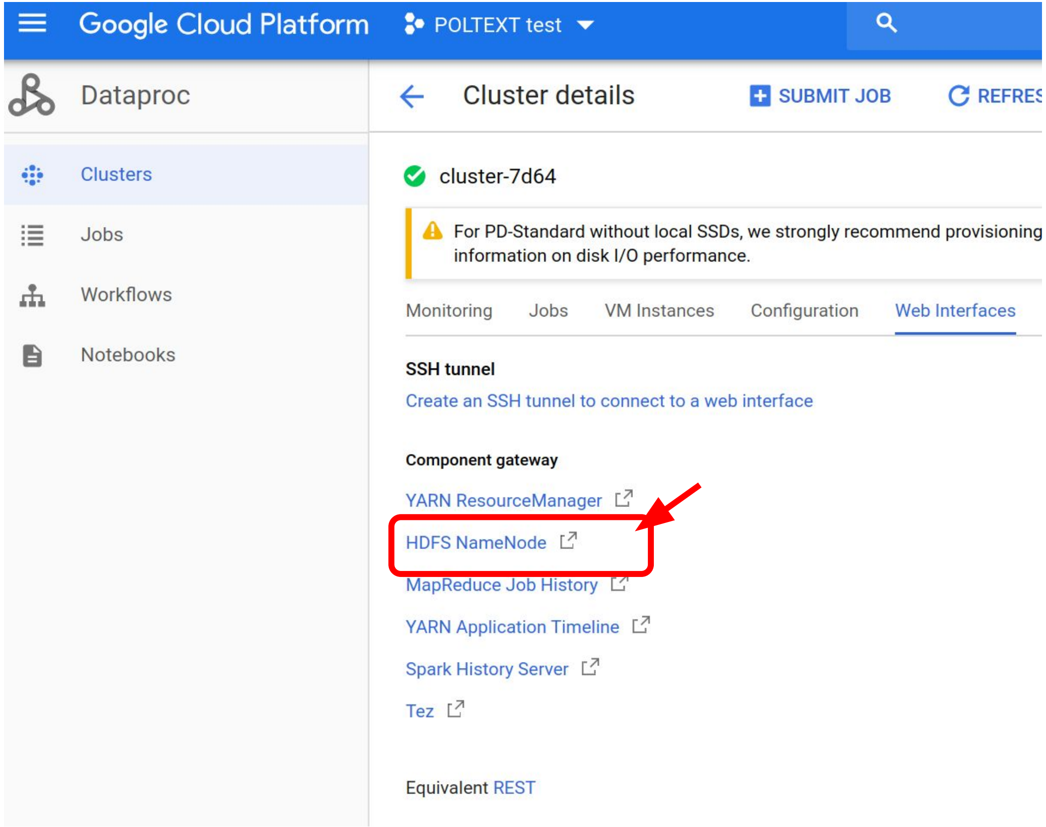
Show 20 entries Search:

ID	User	Name	Application Type	Queue	Application Priority	StartTime	FinishTime	State	FinalStatus	Running Containers	Allocated CPU VCores	Allocated Memory MB	Reserved CPU VCores	Reserved Memory MB	% of Queue	% of Cluster	Progress	Tracking UI	Blacklisted Nodes
application_1567353180718_0001	test	sparklyr	SPARK	default	0	Sun Sep 1 18:53:40 +0200 2019	N/A	RUNNING	UNDEFINED	3	3	4096	0	0	33.3	33.3		ApplicationMaster	

Showing 1 to 1 of 1 entries

First Previous 1 Next Last

This is the YARN cluster manager web UI.
From here we can reach the Spark Master web UI, by pressing ApplicationMaster.



cluster-7d64

For PD-Standard without local SSDs, we strongly recommend provisioning
information on disk I/O performance.

SSH tunnel

Create an SSH tunnel to connect to a web interface

Component gateway

YARN ResourceManager

HDFS NameNode

MapReduce Job History

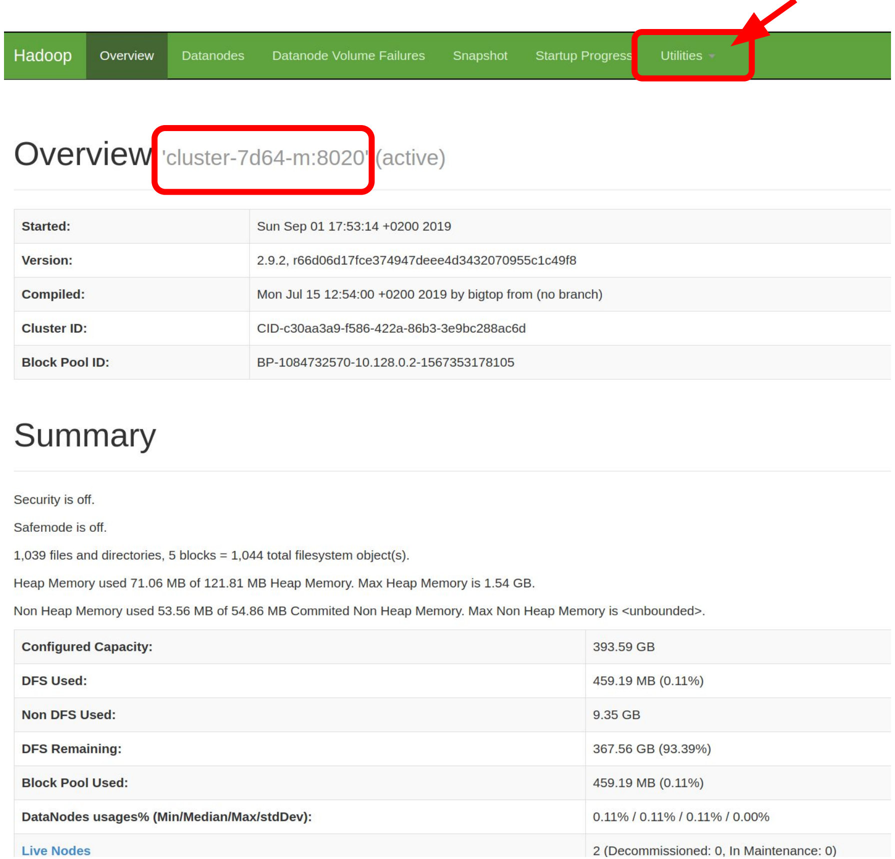
YARN Application Timeline

Spark History Server

Tez

Equivalent REST

Now let’s take a look at our HDFS NameNode.



Overview 'cluster-7d64-m:8020' (active)

Started:	Sun Sep 01 17:53:14 +0200 2019
Version:	2.9.2, r66d06d17fce374947deee4d3432070955c1c49f8
Compiled:	Mon Jul 15 12:54:00 +0200 2019 by bigtop from (no branch)
Cluster ID:	CID-c30aa3a9-f586-422a-86b3-3e9bc288ac6d
Block Pool ID:	BP-1084732570-10.128.0.2-1567353178105

Summary

Security is off.

Safemode is off.

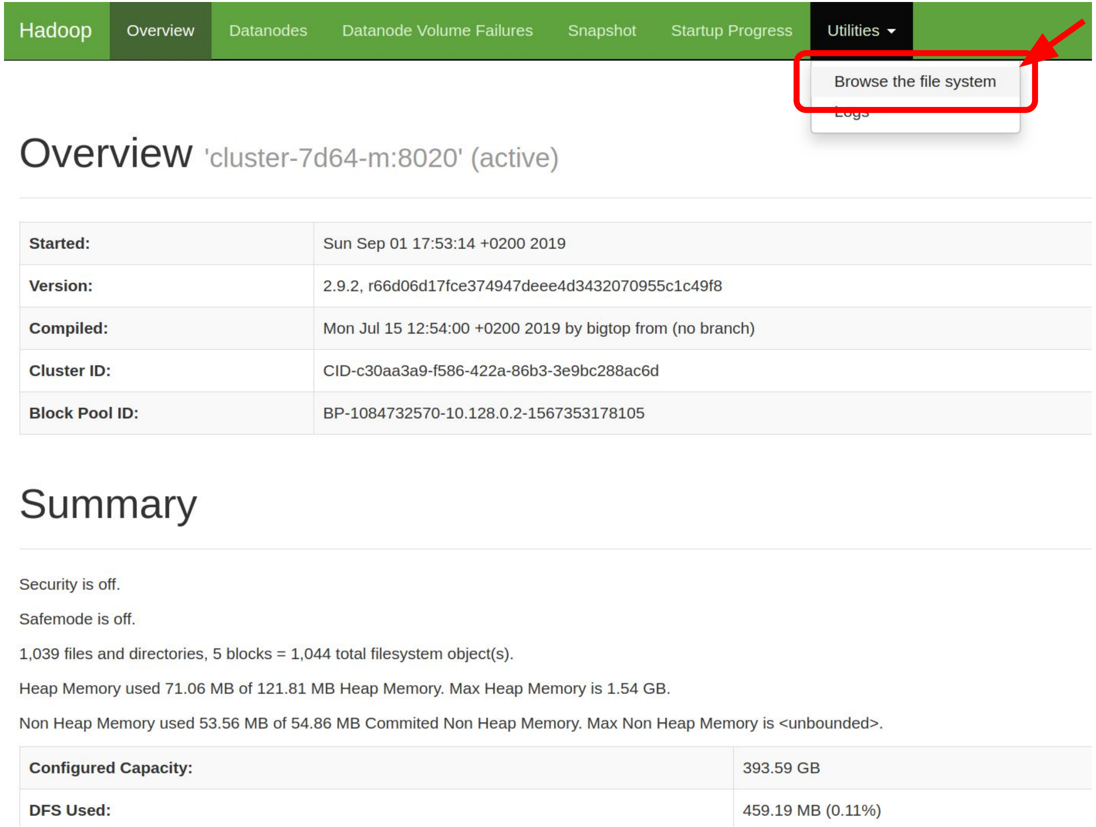
1,039 files and directories, 5 blocks = 1,044 total filesystem object(s).

Heap Memory used 71.06 MB of 121.81 MB Heap Memory. Max Heap Memory is 1.54 GB.

Non Heap Memory used 53.56 MB of 54.86 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	393.59 GB
DFS Used:	459.19 MB (0.11%)
Non DFS Used:	9.35 GB
DFS Remaining:	367.56 GB (93.39%)
Block Pool Used:	459.19 MB (0.11%)
DataNodes usages% (Min/Median/Max/stdDev):	0.11% / 0.11% / 0.11% / 0.00%
Live Nodes	2 (Decommissioned: 0, In Maintenance: 0)

We can see the address of the HDFS NameNode, which we will use to read and write files to the HDFS. We can also move to the browser view, press Utilities.



Overview 'cluster-7d64-m:8020' (active)

Started:	Sun Sep 01 17:53:14 +0200 2019
Version:	2.9.2, r66d06d17fce374947deee4d3432070955c1c49f8
Compiled:	Mon Jul 15 12:54:00 +0200 2019 by bigtop from (no branch)
Cluster ID:	CID-c30aa3a9-f586-422a-86b3-3e9bc288ac6d
Block Pool ID:	BP-1084732570-10.128.0.2-1567353178105

Summary

Security is off.

Safemode is off.

1,039 files and directories, 5 blocks = 1,044 total filesystem object(s).

Heap Memory used 71.06 MB of 121.81 MB Heap Memory. Max Heap Memory is 1.54 GB.

Non Heap Memory used 53.56 MB of 54.86 MB Commited Non Heap Memory. Max Non Heap Memory is <unbounded>.

Configured Capacity:	393.59 GB
DFS Used:	459.19 MB (0.11%)

And select Browse the file system.

For the rest of the topics see:
https://github.com/zkpti/poltext2019-sparktutorial