**What is Apache Spark?**

Spark is a big data solution that has been proven to be easier and faster than Hadoop MapReduce. Spark is an open source software developed by UC Berkeley RAD lab in 2009. Since it was released to the public in 2010.

The main difference between Spark and MapReduce is that Spark runs computations in memory during the later on the hard disk. It allows high-speed access and data processing, reducing times from hours to minutes.

#### What is Pyspark?

Spark is the name of the engine to realize cluster computing while PySpark is the Python's library to use Spark.

## How Does Spark work?

Spark is based on computational engine, meaning it takes care of the scheduling, distributing and monitoring application. Each task is done across various worker machines called computing cluster. A computing cluster refers to the division of tasks. One machine performs one task, while the others contribute to the final output through a different task. In the end, all the tasks are aggregated to produce an output. The Spark admin gives a 360 overview of various Spark Jobs.

**A significant feature of Spark is the vast amount of built-in library, including MLlib for machine learning**. Spark is also designed to work with Hadoop clusters and can read the broad type of files, including Hive data, CSV, JSON, Casandra data among other.

### Why use Spark?

As a future data practitioner, you should be familiar with python's famous libraries: Pandas and scikit-learn. These two libraries are fantastic to explore dataset up to mid-size. Regular machine learning projects are built around the following methodology:

* Load the data to the disk
* Import the data into the machine's memory
* Process/analyze the data
* Build the machine learning model
* Store the prediction back to disk
* Pyspark gives the data scientist an API that can be used to solve the parallel data proceedin problems. Pyspark handles the complexities of multiprocessing, such as distributing the data, distributing code and collecting output from the workers on a cluster of machines.
* Spark can run standalone but most often runs on top of a cluster computing framework such as Hadoop. In test and development, however, a data scientist can efficiently run Spark on their development boxes or laptops without a cluster
* • One of the main advantages of Spark is to build an architecture that encompasses data streaming management, seamlessly data queries, machine learning prediction and real-time access to various analysis.
* • Spark works closely with SQL language, i.e., structured data. It allows querying the data in real time.
* • Data scientist main's job is to analyze and build predictive models. In short, a data scientist needs to know how to query data using SQL, produce a statistical report and make use of machine learning to produce predictions. Data scientist spends a significant amount of their time on cleaning, transforming and analyzing the data. Once the dataset or data workflow is ready, the data scientist uses various techniques to discover insights and hidden patterns. The data manipulation should be robust and the same easy to use. Spark is the right tool thanks to its speed and rich APIs.

## Spark Context

SparkContext is the internal engine that allows the connections with the clusters. If you want to run an operation, you need a SparkContext.

import pyspark

from pyspark import SparkContext

sc =SparkContext()

Now that the SparkContext is ready, you can create a collection of data called RDD, Resilient Distributed Dataset. Computation in an RDD is automatically parallelized across the cluster.

nums= sc.parallelize([1,2,3,4])

You can access the first row with take

nums.take(1)

[1]

You can apply a transformation to the data with a lambda function. In the example below, you return the square of nums. It is a map transformation

squared = nums.map(lambda x: x\*x).collect()

for num in squared:

print('%i ' % (num))

1

4

9

16

## SQLContext

A more convenient way is to use the DataFrame. SparkContext is already set, you can use it to create the dataFrame. You also need to declare the SQLContext

SQLContext allows connecting the engine with different data sources. It is used to initiate the functionalities of Spark SQL.

from pyspark.sql import Row

from pyspark.sql import SQLContext

sqlContext = SQLContext(sc)

Let's create a list of tuple. Each tuple will contain the name of the people and their age. Four steps are required:

**Step 1)** Create the list of tuple with the information

[('John',19),('Smith',29),('Adam',35),('Henry',50)]

**Step 2)** Build a RDD

rdd = sc.parallelize(list\_p)

**Step 3)** Convert the tuples

rdd.map(lambda x: Row(name=x[0], age=int(x[1])))

**Step 4)** Create a DataFrame context

sqlContext.createDataFrame(ppl)

list\_p = [('John',19),('Smith',29),('Adam',35),('Henry',50)]

rdd = sc.parallelize(list\_p)

ppl = rdd.map(lambda x: Row(name=x[0], age=int(x[1])))

DF\_ppl = sqlContext.createDataFrame(ppl)

If you want to access the type of each feature, you can use printSchema()

DF\_ppl.printSchema()

root

|-- age: long (nullable = true)

|-- name: string (nullable = true)