ML with ApacheSpark_Basics

As per Apache Spark official website spark has below advantages over other available tools (hadoop, mapreduce) in market:

- **Speed**: Run workloads 100x faster [for both batch and streaming data]
- **Ease of Use**: Write applications quickly in Java, Scala, Python, R, and SQL.
- **Generality**: Combine SQL, streaming, and complex analytics.
- **Runs Everywhere**: Spark runs on Hadoop, Apache Mesos, Kubernetes, standalone, or in the cloud. It can access diverse data sources.

Intro

Spark is a distributed computing platform for working with Big Data. It achieves efficiency by distributing data and computation across a cluster of computers. Spark is currently the best framework for cluster computing. Spark transparently handles the distribution of compute tasks across a cluster. Spark operations are fast and it also allows you to focus on the analysis rather than worry about technical details.

Spark is popular because of below reasons:

1. Much faster than other big data technology like Hadoop because it does most processing in memory
2. Developer friendly interface which hides much of the complexity of distributed computing [Spark has a high-level API, which conceals a lot of complexity.]
3. Compute across a distributed cluster.

Architecture

- Each cluster contains one or more nodes [each node is a computer with cpu, ram and physical storage]
- Cluster manager allocates resources and coordinates activities across the cluster
- Every app running on spark cluster has a driver program
- Using spark api driver communicates with cluster manager which in turn distribute works to the nodes
- On each node spark launches an executor process which persists for the duration of the application
- Work is divided into tasks which are simply unit of computation
- Executor runs tasks in multiple threads across the cores in a node

Sparks sets up all above infrastructure by itself and handle all interaction within the cluster. we don’t need to worry much about it. However it's still useful to know how does it work under the hood.

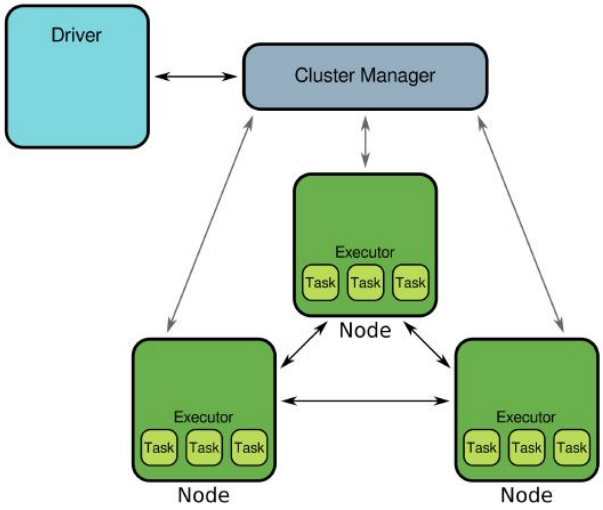

image source: Datacamp

Connecting with Spark:

- The connection to spark is established by the driver which can be written either in Java, Scala, Python or R.
- Java is relatively verbose which requires which requires a lot of code even for simple task.
- Scala, Python and R are high level languages which can accomplish much with small amount of code. They also offer REPL [read, evaluate, print loop] which is crucial for interactive development.
- python does not talk natively to spark hence we have to import pyspark which make spark functionality available in python interpreter.

Remote and Local cluster connection:

Before connecting with spark we need to tell spark where the cluster is located.

```
1.Remote cluster: We can connect to a remote cluster where we need to specify the
 URL for remote cluster.
    URL format: spark://<IP address | DNS name>:<port>.
    Default port is 7077 but we must specify this explicitly.
2.Local cluster [everythimng happens on a single computer]:
    local – only 1 core;
    local[2] – 2 cores; or
    local[*] – all available cores.
```

Installation

- pip install pyspark

Import and check version

In [3]:

```
import pyspark
pyspark.__version__
```

Out[3]:

```
'2.4.3'
```

Creating a spark session

In [8]:

```
# Import the PySpark module
from pyspark.sql import SparkSession
```

Create a local cluster using a SparkSession builder

- specify location of cluster using "master" method
- assign a name using "appName" method. This is optional.
- "getOrCreate" method either will create a new session or return an existing session object

In [ ]:

```
# Create SparkSession object
spark = SparkSession.builder.master('local[*]').appName("test").getOrCreate()
```

Close session

When you are done with the session it is a good practice to close the session

In [9]:

```
# Terminate the cluster
spark.stop()
```

The SparkSession class has a **version** attribute which gives the version of Spark.

In [4]:

```
# What version of Spark?
print(spark.version)
```

```
2.4.3
```

The session object will now allow us to load data into Spark.

Reference:

- Apache Spark (https://spark.apache.org/)
- towardsdatascience (https://towardsdatascience.com/deep-learning-with-apache-spark-part-1-6d397c16abd)
- Datacamp (https://www.datacamp.com/community/tutorials/apache-spark-tutorial-machine-learning)