## Fundamentals of Big Data

**BIG DATA FUNDAMENTALS WITH PYSPARK** 



Upendra Devisetty
Science Analyst, CyVerse



#### What is Big Data?

 Big data is a term used to refer to the study and applications of data sets that are too complex for traditional data-processing software - Wikipedia



#### The 3 V's of Big Data

- Volume, Variety and Velocity
- Volume: Size of the data
- Variety: Different sources and formats
- Velocity: Speed of the data



#### Big Data concepts and Terminology

- Clustered computing: Collection of resources of multiple machines
- Parallel computing: Simultaneous computation on single computer
- Distributed computing: Collection of nodes (networked computers) that run in parallel
- Batch processing: Breaking the job into small pieces and running them on individual machines
- Real-time processing: Immediate processing of data



#### Big Data processing systems

- Hadoop/MapReduce: Scalable and fault tolerant framework written in Java
  - Open source
  - Batch processing
- Apache Spark: General purpose and lightning fast cluster computing system
  - Open source
  - Both batch and real-time data processing
- Note: Apache Spark is nowadays preferred over Hadoop/MapReduce

#### Features of Apache Spark framework

- Distributed cluster computing framework
- Efficient in-memory computations for large data sets
- Lightning fast data processing framework
- Provides support for Java, Scala, Python, R and SQL

#### **Apache Spark Components**

Spark SQL MLIib Machine Learning

GraphX

Spark Streaming

RDD API
Apache Spark Core

#### Spark modes of deployment

- Local mode: Single machine such as your laptop
  - Local model convenient for testing, debugging and demonstration
- Cluster mode: Set of pre-defined machines
  - Good for production
- Workflow: Local -> clusters
- No code change necessary

# Coming up next - PySpark

BIG DATA FUNDAMENTALS WITH PYSPARK



## PySpark: Spark with Python

BIG DATA FUNDAMENTALS WITH PYSPARK



Upendra Devisetty
Science Analyst, CyVerse



#### Overview of PySpark

- Apache Spark is written in Scala
- To support Python with Spark, Apache Spark Community released PySpark
- Similar computation speed and power as Scala
- PySpark APIs are similar to Pandas and Scikit-learn

#### What is Spark shell?

- Interactive environment for running Spark jobs
- Helpful for fast interactive prototyping
- Spark shells allow interacting with data on disk or in memory
- Three different Spark shells:
  - Spark-shell for Scala
  - PySpark-shell for Python
  - SparkR for R

#### PySpark shell

- PySpark shell is the Python-based command line tool
- PySpark shell allows data scientists interface with Spark data structures
- PySpark shell support connecting to a cluster

#### **Understanding SparkContext**

- SparkContext is an entry point into the world of Spark
- An entry point is a way of connecting to Spark cluster
- An entry point is like a key to the house
- PySpark has a default SparkContext called sc

<sup>&</sup>lt;sup>1</sup> https://www.datacamp.com/cheat-sheet/pyspark-cheat-sheet-spark-in-python



#### Inspecting SparkContext

Version: To retrieve SparkContext version

sc.version

#### 2.3.1

• Python Version: To retrieve Python version of SparkContext

sc.pythonVer

3.6

Master: URL of the cluster or local string to run in local mode of SparkContext

sc.master

local[\*]

<sup>1</sup> https://www.datacamp.com/cheat-sheet/pyspark-cheat-sheet-spark-in-python



#### Loading data in PySpark

• SparkContext's parallelize() method

```
rdd = sc.parallelize([1,2,3,4,5])
```

• SparkContext's textFile() method

```
rdd2 = sc.textFile("test.txt")
```

<sup>&</sup>lt;sup>1</sup> https://www.datacamp.com/cheat-sheet/pyspark-cheat-sheet-spark-in-python



## Let's practice

BIG DATA FUNDAMENTALS WITH PYSPARK



# Use of Lambda function in python - filter()

BIG DATA FUNDAMENTALS WITH PYSPARK



**Upendra Devisetty**Science Analyst, CyVerse



#### What are anonymous functions in Python?

- Lambda functions are anonymous functions in Python
- Very powerful and used in Python. Quite efficient with map() and filter()
- Lambda functions create functions to be called later similar to def
- It returns the functions without any name (i.e anonymous)
- Inline a function definition or to defer execution of a code

#### Lambda function syntax

The general form of lambda functions is

```
lambda arguments: expression
```

Example of lambda function

```
double = lambda x: x * 2
print(double(3))
```

6

#### Difference between def vs lambda functions

Python code to illustrate cube of a number

```
def cube(x):
    return x ** 3

g = lambda x: x ** 3

print(g(10))
print(cube(10))
```

```
1000
1000
```

- No return statement for lambda
- Can put lambda function anywhere

#### Use of Lambda function in Python - map()

- map() applies a function to all items in the input list
- General syntax of map()

```
map(function, list)
```

Example of map()

```
items = [1, 2, 3, 4]
list(map(lambda x: x + 2 , items))
```

```
[3, 4, 5, 6]
```

#### Use of Lambda function in python - filter()

- filter() function takes a function and a list and returns a new list for which the function evaluates as true
- General syntax of filter()

```
filter(function, list)
```

• Example of filter()

```
items = [1, 2, 3, 4]
list(filter(lambda x: (x%2 != 0), items))
```

```
[1, 3]
```

## Let's practice

BIG DATA FUNDAMENTALS WITH PYSPARK

