MACHINE LEARNING SUPERVISIONATO CON APACHE SPARK
HANDS-ON CODELAB

IAML– Hands on Codelab 16/07/2019 - Roma
Machine Learning Supervisionato con Apache Spark
Speaker Valerio Morfino

1

VALERIO MORFINO

Head of Big Data & Analytics at DbServices srl

Valerio Morfino ingegnere informatico è Head of Big Data & Analytics presso DB Services. Nel corso della propria carriera ha lavorato in società di consulenza, università ed aziende occupandosi di consulenza, formazione, ricerca, direzione di progetti. E' autore di articoli e relatore in conferenze sui temi web e-commerce, machine learning e big data.

2

Summary

- Why Big Data?
- HDFS & Map Reduce
- Apache Spark
- Spark in the Cloud: Databricks
- Case Study Introduction
- Hands on!

Big Data

Why Big Data ?

How many bytes to store DNA?

- about **125 Mb,** to store variations from “reference” genome
- about **700 Mb,** storing a plain text sequence of nucleotides such as "AGCTGGCGGT" without additional informations
- about **200 Gb,** storing the output of a sequencer in a format such as FASTQ (with all metadata)

Source: https://www.linkedin.com/pulse/how-many-bytes-we-need-store-dna-all-peoples-world-valerio-morfino/

How many bytes to store DNA?

Population of Italy: **60.5 Millions**

7.56 Petabyte

42.35 Petabyte

12.10 Exabyte

…Up to 1.52 Zettabyte for the whole world!

Source: https://www.linkedin.com/pulse/how-many-bytes-we-need-store-dna-all-peoples-world-valerio-morfino/

How many bytes to store DNA?

Of course it is needed software, a lot of space and a lot of CPUs to analyze and query this data, but **thanks to Big Data, Artificial Intelligence techniques and Cloud Computing,** now we have the chance to **face problems like this,** where **very large amounts of data are involved**

Hadoop and Map Reduce

Ok, but…

How ?

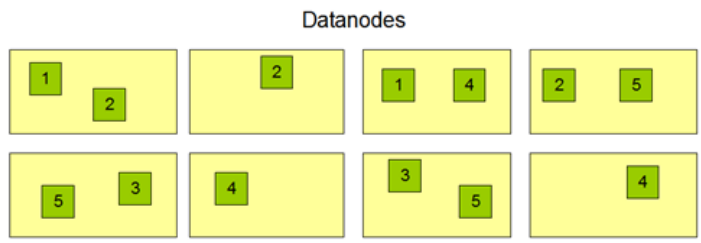
How many bytes to store DNA?

Two big problems:

- **How to store (in a secure way) such a large amount of data?**
- **How to process such data?**

- **We are facing the main Big Data problems!**

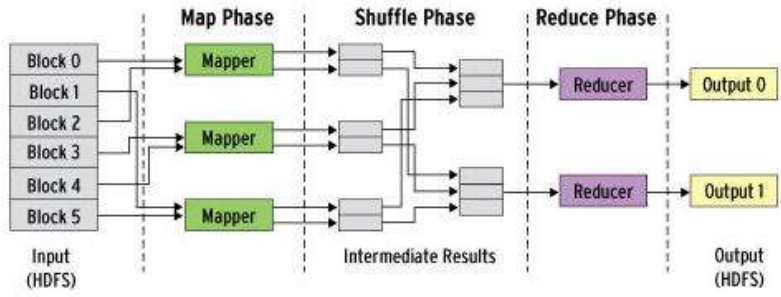
HDFS



Source: https://hadoop.apache.org/docs/current/hadoop-project-dist/hadoop-hdfs/HdfsDesign.html

- Hadoop File System, a distributed filesystem
- Each file is broken into small parts (chunks), replicated
- NameNode, know where the pieces are
- DataNode, store file parts

Map Reduce Paradigm



Source: http://www.admin-magazine.com/HPC/Articles/MapReduce-and-Hadoop

- Map jobs read a block of data and produce key-value pairs
- Reducer jobs receives key-value pairs from multiple map jobs, sorted by key and produce output
- Each partial result is stored on HDFS

Apache Spark

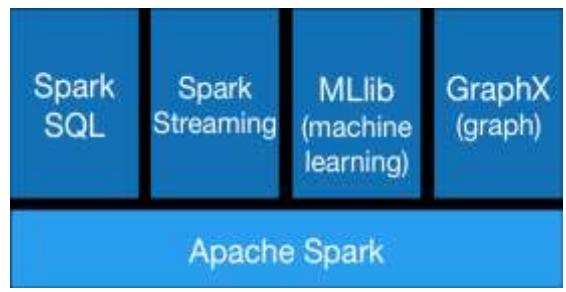
Apache Spark

Apache Spark

- A distributed cluster based general engine for big data processing
  - Fully integrated with Hadoop ecosystem
  - Available both in local and in cloud environments
  - Clusters of hundreds or even thousands of nodes
  - Up to 100X faster than Hadoop **Map Reduce**
  - **Resilient** thanks to lineage and distributed file (e.g. HDFS)
    - This is important for Big Data and long processing tasks on big clusters and hardware, software or networks can fail!

Apache Spark

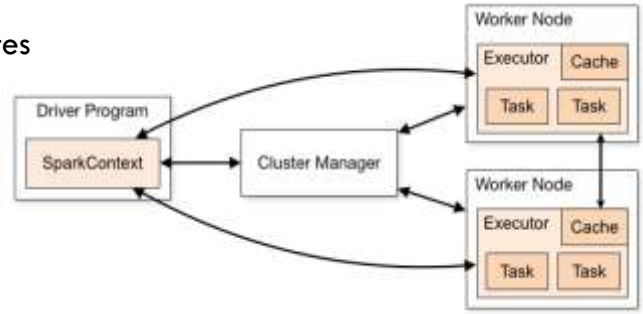
- High-level APIs accessible in Java, **Scala,** Python and R



- The MLlib library is rich of efficient parallel implementation of Machine learning algorithms

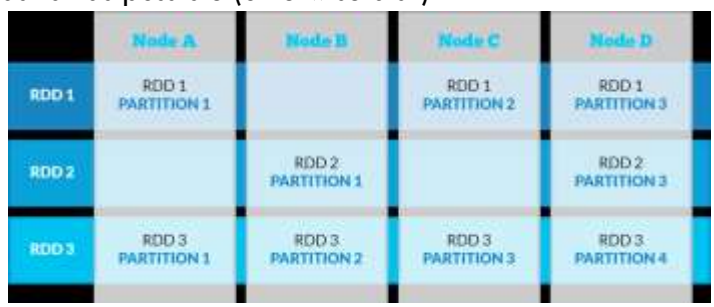
Spark Cluster configurations

- Several Cluster configurations:
  - Stand Alone
  - Hadoop Yarn
  - Mesos
  - Kubernetes



RDDs to store Large datasets

- Resilient, i.e. fault-tolerant thanks to RDD lineage graph, able to recompute missing or damaged partitions
- Distributed, with data residing on multiple nodes in a cluster
- Dataset is a collection of partitioned data stored in memory as far as possible (otherwise disk)



Spark SQL, DataFrames and Datasets

- **Spark SQL** is a Spark module for structured data processing.
- A **Dataset** is a distributed collection of data. Only supported by Java and Scala API.
- A **DataFrame** is a *Dataset* organized into named columns. **It is conceptually equivalent to a table in a relational database or a data frame in R or Python,** but with richer optimizations under the hood
- Dataset and Dataframe are internally represented as **RDD** but executed with some optimizations!

Mllib - Spark's machine learning library

- ML Algorithms: common learning algorithms such as **classification, regression, clustering,** and **collaborative filtering**
- **Featurization:** feature extraction, transformation, dimensionality reduction, and selection
- **Pipelines:** tools for constructing, evaluating, and tuning ML Pipelines
- **Persistence:** saving and load algorithms, models, and Pipelines
- **Utilities:** linear algebra, statistics, data handling, etc.
- **Text Manipulations:** Tokenization, Common Word Removing, Word combinations, Word2Vec

Note: As of Spark 2.0, DataFrame-based API is primary API (package spark.ml). The MLlib RDD-based API is now in maintenance mode (package spark.mllib)

Useful links

- https://spark.apache.org/docs/latest/
- https://spark.apache.org/docs/latest/ml-guide.html
- https://spark.apache.org/docs/latest/ml-classification-regression.html
- https://docs.databricks.com/getting-started/index.html

19

Data Bricks

Databricks in a nutshell

- www.databricks.com
- From the creators of Apache Spark
- Databricks unifies data science and engineering across the Machine Learning lifecycle from data preparation to experimentation and deployment of ML applications.
- Databricks Platform (commercial version)
  - For businesses looking for a zero-management cloud platform built around Apache Spark
- Community Edition

Databricks – community edition

- Single cluster limited to 6GB and no worker nodes
- Basic notebook without collaboration
- Limited to 3 max users
- Public environment to share your work
- Create a community profile: https://databricks.com/try-databricks

Upload and download files

- Upload and download via API or Interface to DBFS (Data Bricks File System)
  - https://docs.databricks.com/user-guide/importing-data.html
- Upload via web interface

- Download file via web interface:
  - *Filename: /FileStore/tables/titanic.csv*
  - *Link*: https://community.cloud.databricks.com/files/tables/titanic.csv?o=XXXXXXXX

  XXXXXXXX is in the URL of yout current Databricks session

23

Let's take a look at…

- Create a Cluster
- Upload Data
- Create a Notebook
- Execute shell command, sql, and much more in notebook:
  - https://docs.databricks.com/user-guide/notebooks/notebook-use.html
- Access Spark consolle

24

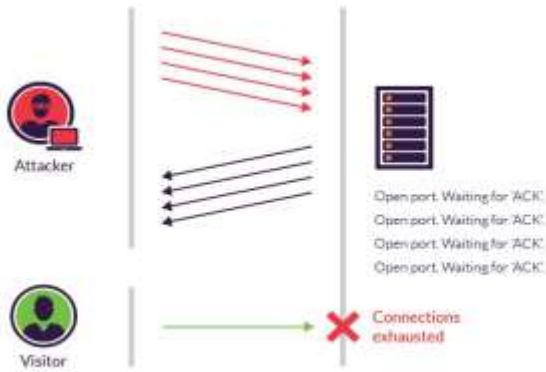
CASE STUDY 1:
SYD-DOS attack prediction

Attacchi informatici

- Possono minare:
  - Riservatezza
  - Integrità
  - Disponibilità

- Gli attacchi DOS – Denial of Service minano la Disponibilità
- L'attacco SYN-DOS (detto anche SYN-Flood) mina la disponibilità saturando le connessioni TCP/IP del server

SYN-DOS Attack



https://www.imperva.com/learn/application-security/syn-flood/

1. Client requests connection by sending SYN (synchronize) message to the server.
2. Server acknowledges by sending SYN-ACK (synchronize-acknowledge) message back to the client.
3. Client responds with an ACK (acknowledge) message, and the connection is established.

SYN flood attack

- The attacker sends repeated SYN packets to every port on the targeted server, often using a fake IP address. The server receives multiple, apparently legitimate requests to establish communication. It responds to each attempt with a SYN-ACK.
- The malicious client either does not send the expected ACK, or—if the IP address is spoofed—never receives the SYN-ACK.
- The server under attack wait for SYN-ACK packet for some time (timeout). During this time, the server cannot close the connection and another SYN packet arrive. This leaves an increasingly large number of connections half-open. As the server's connection overflow tables fill, service to legitimate clients will be denied and the server may even malfunction or crash.

Dataset & Reference

- Dataset Description
  - 115 features (Double)
  - 1 Label (String)
  - 11.000 total samples (10.000 normal + 1.000 attack)
- Features contains statistics which are used to implicitly describe the current state of the channel
- Data came from IP-Cameras
- The statistics are generated by a Feature Extractor

- Syn-Dos
- Paper: https://arxiv.org/pdf/1802.09089.pdf
- Full Dataset: https://drive.google.com/drive/folders/1kmoWY4poGWfmmVSdSu-r_3Vo84Tu4PyE

29

SYNDOS - Labs

- Train a DecisionTreeClassifier and compare the accuracy
- Print the Decision Tree and parameters importance
- Train a Multilayer Perceptron
- K-fold validation
- Model Tuning

30

CASE STUDY 2:
DNA splicing site prediction

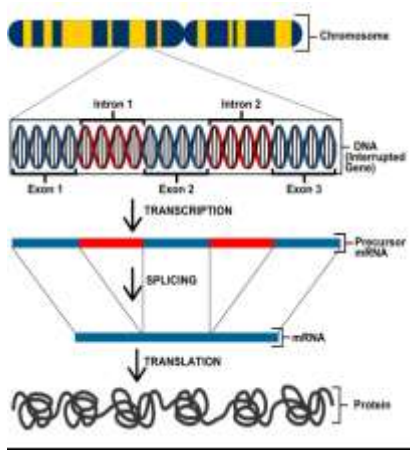
DNA splicing site prediction

- We deal with a bioinformatic problem: **splicing site prediction**
- Useful for:
  - Biological Research (identification of Intron-Exon boundaries)
  - Medical research (to understand human variation on splicing and its effect on human diseases)
  - Personalized medicine

Biological Background

- DNA is a linear molecule composed of four small molecules called nucleotide bases: adenine (A), cytosine (C), guanine (G), and thymine (T).
- Segments of DNA that carry genetic information are called genes.
- The genes in DNA encode protein molecules according to the flow known as "The Central Dogma": DNA → mRNA → Protein.

Biological Background II



- Most of eukariotic genes have their coding sequences – **exons**- interrupted by non-coding sequences - **introns**.
- The interruption points between exon-intron (EI or donor) and intron-exon (IE or acceptor) are called "splicing sites". During the splicing process introns are removed
- The DNA splicing site prediction problem deals with individuating those regions.

Splicing site problem in ML terms

- Given a sequence of DNA (e.g. 60 nucleotides) : AGTGTCCAGTCATG…GT…GAACGTAAGTAAGA
- We wish to classify each sequence as:
  - Containing a splicing site in the middle
  - Not containing a splicing site in the middle
- Binary single one-value encoding (one hot encoding):

$$A \rightarrow 1000; \quad C \rightarrow 0100; \quad G \rightarrow 0010; \quad T \rightarrow 0001$$

  - With 60 nucteotides we have 240 binary attributes

$$f_c : \{0,1\}^{240} \rightarrow \{0,1\}$$

Supervised Machine learning recipe

- **Ingredients:**
  - A labelled set of data

    In this case four files:

    pos_training, neg_training, pos_test, neg_test
  - A learning algorithm (e.g. Decision tree, SVM, Random Forest, Multi Layer Perceptron, …)
- **Preparation:**

1. Load Dataset and assign a label

   AGTGTCCAGTCATG…GT…GAACGTAAGTAAGA,1
2. Encode features (OneHot Encoder)

   0,0,0,1,0,0,0,1,0,…,0,0,1,0,0,0,0,1,0,1,0,0,1,0,0,1,0,0,1 One Hot

Note: The last field is the label: 1-> Splicing site; 0-> not splicing site

Supervised Machine learning cookbook

3. Split the Input Dataset in:
   - Training set (about 70-80%)
   - Test set (about 20-30%)
4. Assemble features in a Vector

   0,0,0,1,0,0,0,1,0,…,0,0,1,0,0,0,0,1,0,1,0,0,1,0,0,1,0,0,1

   features, label

   [0,0,0,1,0,0,0,1,0,…,0,0,1,0,0,0,0,1,0,1,0,0,1,0,0,1,0,0],1
5. Train a Model
6. Test the model on Test set (tune and refine…)
7. Ready to classify new unlabbelled data!

THANK YOU!

valerio.morfino@dbservices.it
https://it.linkedin.com/in/valerio-morfino