H_2O.ai

Bauhaus-Universität Weimar | SS 16

Big Data Architectures for Machine Learning and Data Mining

Patrick Saad

H2O

Outline

Short talks recap

Overview

Machine Learning Algorithms

H2O Flow + demo

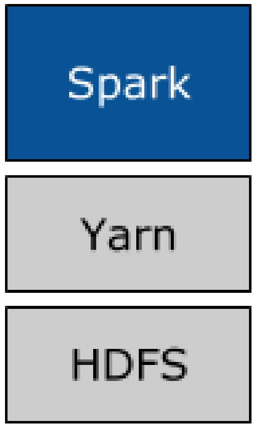
Sparkling Water

H2O

Short talks recap

Apache Spark:

- is open source!
- is way faster than Hadoop (**memory** vs **hard-disk** speeds)
- integrates with Hadoop (reads/writes to **Hadoop HDFS**)
- data filtering with **Spark SQL**
- has **Mllib** as a library of common machine learning (**ML**) algorithms
    - > **spark.ml** package for constructing **ML** pipelines



H2O
Overview

- **(Not an Apache)** open source ML engine with a main aim that's "*... bringing AI to business through software*" - http://www.h2o.ai
    - Apache targets expert users? ML still not as "popular" and accessible for businesses as other applications of computer science (web dev, app dev)

- Apply math and predictive analytics on huge datasets (can handle 100GB, currently working on 1TB)
    - fast **in-memory** distributed parallel processing (same as Spark)

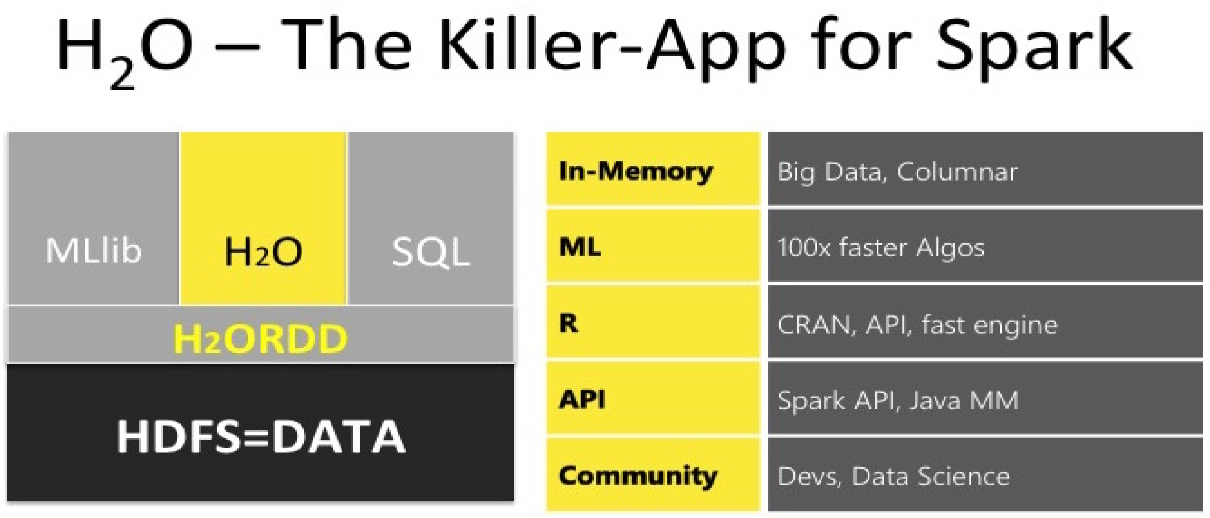
- Works with Java, Python, R, Scala, JSON and through APIs

- Ready-to-use ML algorithms (same as Spark Mllib)

H2O

Overview

- Flow UI is one of the difference makers
    - **No advanced ML knowledge** needed
    - Point-and-click
- Spark + H2O = Sparkling Water
- Run on Hadoop YARN


H_2O – The Killer-App for Spark

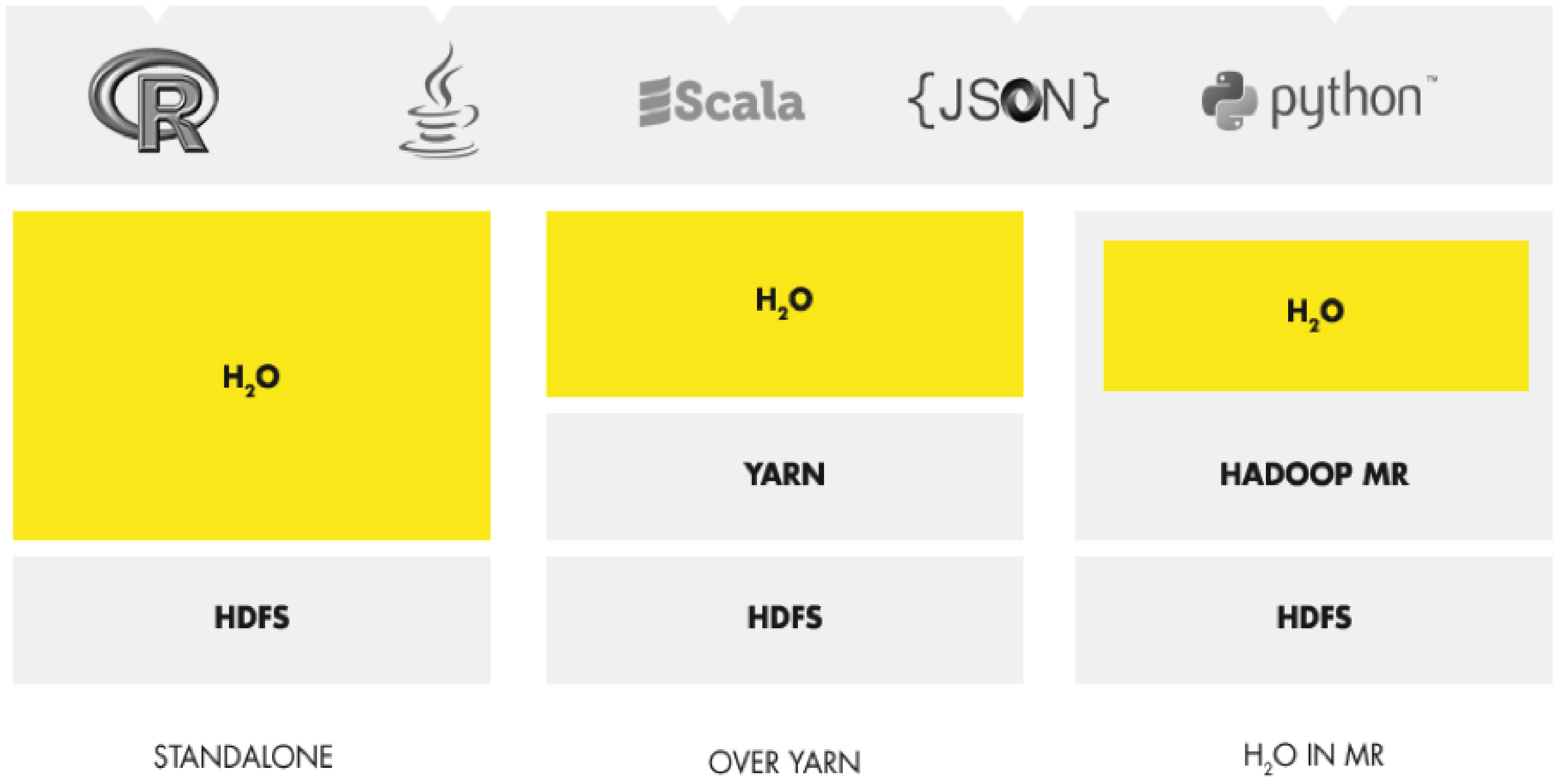
In-Memory	Big Data, Columnar
ML	100x faster Algos
R	CRAN, API, fast engine
API	Spark API, Java MM
Community	Devs, Data Science

Credits: h2o.ai

H2O

Overview

Summary



Credits: hortonworks.com

H2O

ML Algorithms

Classification

- Generalized Linear Models (**GLM**)
  - linear regression supporting also non-normal distribution models (Poisson, binomial, etc...)
- Decision Trees (*see Image 1*)
- Gradiant Boosting (**GBM**)
  - prediction model is an group of usually decision trees
- Naïve Bayes

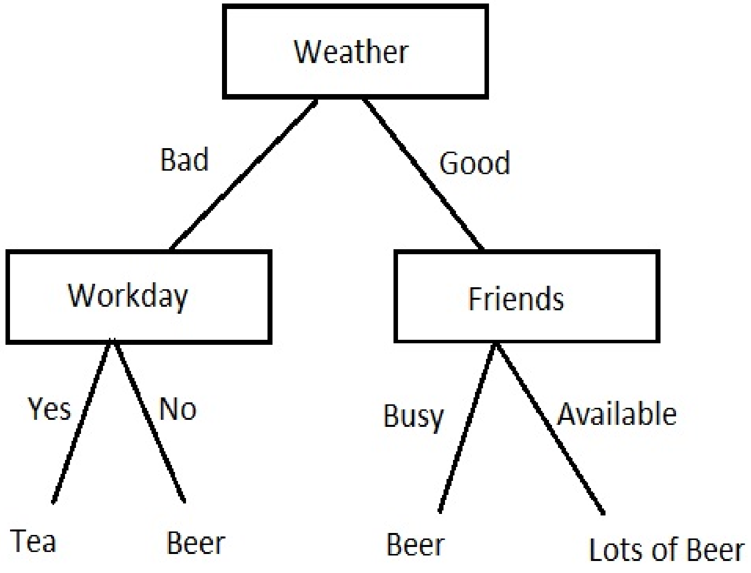


Image 1

H2O
Machine Learning Algorithms (ML)

Regression

- GLM
- GBM

Clustering

- K-Means
    - defines one centroid per cluster K and associate data points to the closest centroid

Hyperparameter tuning / optimization

- Anomaly Detection (outliers)
- Grid Search (finds parameters which produce optimate learning performance)

Deep Learning (with no complicated configuration)

H2O
Flow UI

Web-based environment with code execution, model building, data annotation and data visulization and presentation. No-programming knowledge required.

Running a demo using H2O v. 3.8.2.2 **| General Guideline**

1- Download http://www.h2o.ai/download/h2o/desktop

```
unzip h2o-3.8.2.2.zip
cd h2o-3.8.2.2
java -jar h2o.jar
```

2- **Go to** http://localhost:54321

3- **Import** dataset by writing the importFiles command (**OR** point-and-click)

4- **Parse** dataset using interface (point-and-click)

5- **Build** model by selecting one of the available algorithms and its settings (point-and-click)

6- **Predict** scores using built model (point-and-click)

H2O

Flow UI

Live Demo

a- **Algorithm**: K-means

b- **Dataset**: Seeds dataset: http://archive.ics.uci.edu/ml/datasets/seeds
http://archive.ics.uci.edu/ml/machine-learning-databases/00236/seeds_dataset.txt

i. **Abstract**: "Measurements of geometrical properties of kernels belonging to three different varieties of wheat. A soft X-ray technique and GRAINS package were used to construct all seven, real-valued attributes."

ii. **Instances**: 210 **Attributes**: 7 (real-valued continuous)

iii. **Attribute information**:

1. area A
2. perimeter P
3. compactness C = 4*pi*A/P^2
4. length of kernel
5. width of kernel
6. asymmetry coefficient
7. length of kernel groove

H2O
Flow UI

Live Demo

c- Running:

i. Run the h2o.jar file, go to http://localhost:54321 and create a **New** notebook.

ii. Import the dataset:

http://s3.amazonaws.com/h2o-public-test-data/smalldata/flow_examples/seeds_dataset.txt

iii. Use the Flow UI controls to continue (commands are also possible):

- **>> Parse these files...**
- Leave default parse settings:

  Column types **Numeric**, Parse **CVS**, Seperator **HT '\t'**, etc ...

  >> Parse

>> View (parse output is a .hex file)

H2O
Flow UI

Live Demo

>> Split...
- training 75%, testing 25%

>> Build Model...
- Select **K-means** as the algorithm
- Select the training frame (.hex file)
- **ignored_columns**: C8 (the class in the dataset)
- Specify the number of clusters **k***
- **init**: PlusPlus (initial center randomly chosen, random subsequent centers are weighted so the points furthest away from the initial center are chosen)
- uncheck **Standarize** columns

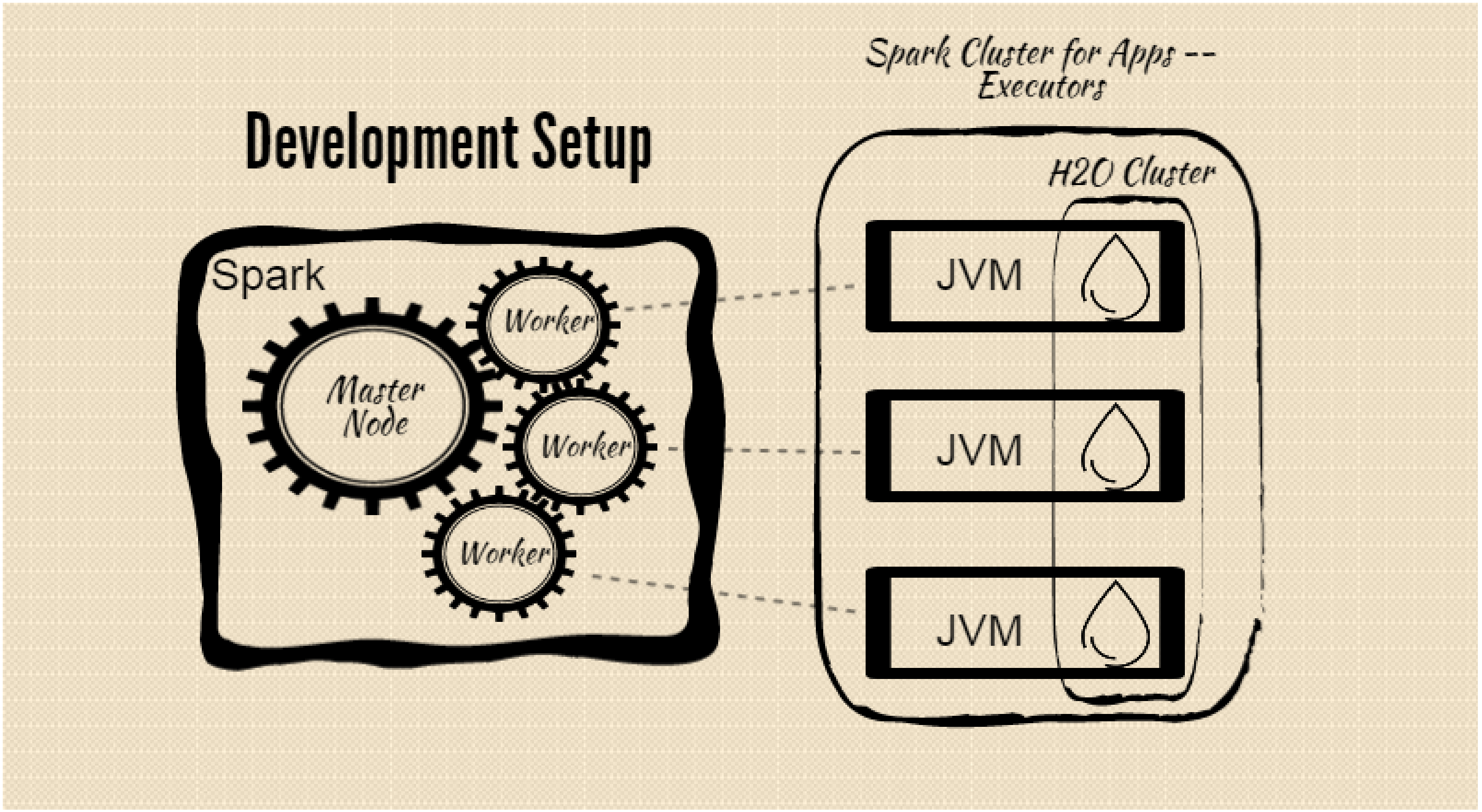
>> Build Model

- Actions **>> View**

>> Predict...
- Choose testing frame

H2O

Sparkling Water



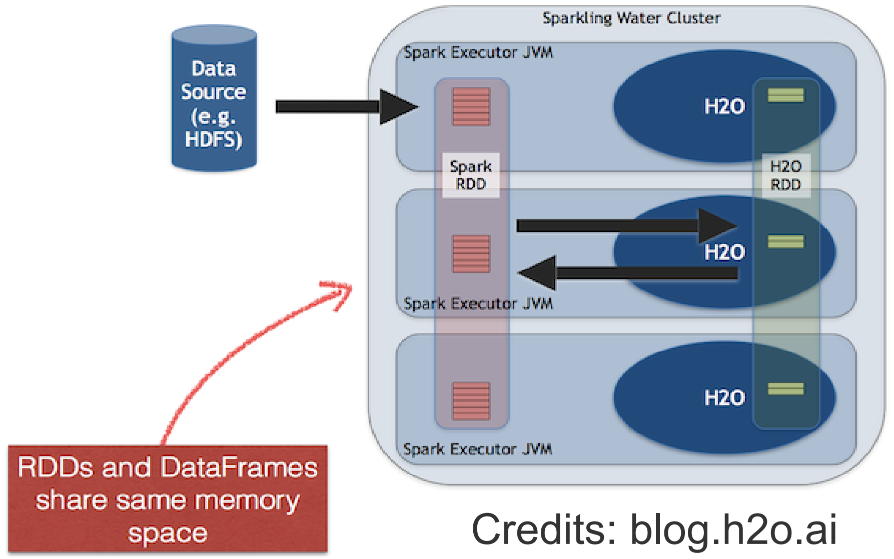
Credits: h2o.ai

H2O

Sparkling Water

Data Distribution

- H2O as an app for Spark
    - use its ML algorithms with Spark
    - share data structures
- Use Scala or Python to build models
- All H2O features are included (e.x: Flow UI)



Credits: blog.h2o.ai

Running a demo –
using Spark v. 1.4.1 and Sparkling Water v. 1.4.14

1-

```
unzip sparkling-water-1.4.14.zip
cd sparkling-water-1.4.14
```

2- Run example by executing **bin/run-example**

3- Run the Spark shell by executing **bin/sparkling-shell**
    3.1 – **openFlow** to open H2O's Flow UI in the browser
    3.2 – **openSparkUI** to open the Spark UI in the browser
    3.3 – Write Scala code within the shell ...

H2O

References

H2O Docs http://h2o.ai/docs

H2O on Github https://github.com/h2oai

H2O on Google Groups mailing list https://groups.google.com/forum/#!forum/h2ostream