sparklyR: R and Spark

An Introduction

Marck Vaisman

November 26, 2016

Disclaimer

The opinions expressed in this presentation are mine and they do not reflect in any way those of the organizations which I am affiliated with.

Show of hands

- Who has experience with Hadoop?

Show of hands

- Who has experience with Hadoop?
- Who has experience with Spark?

Show of hands

- Who has experience with Hadoop?
- Who has experience with Spark?
- Who has experience with Scala?

Show of hands

- Who has experience with Hadoop?
- Who has experience with Spark?
- Who has experience with Scala?
- Who **loves** R?

Agenda

- About Spark
- Introduce sparklyr and compare it to SparkR
- Show how to install and configure so everything works together nicely on AWS EMR
- Quick Demo

Spark

- An Apache project providing **lightning fast cluster computing**
- Open source cluster computing framework
- Works with data in memory as opposed to batch io from disk
- Developed in Scala
- Provides APIs in Scala, Java, Python and R
- Can run on clusters managed by YARN, Mesos or Standalone
- Lazy evaluation: no computations are performed until an action is taken or the data is collected back to the driver (master)
- Awesome, but finicky

sparklyr

- A new package developed by the RStudio team (enough said)
- Provides a complete dplyr interface to Spark RDDs
- Transforms dplyr verbs into SparkSQL commands run that act on a Spark DataFrame
- **Not** a replacement to SparkR
- Lowers barrier to entry into Spark for R users

Easy Button!

From this (native Scala on spark-shell)

```
val queries = sc.textFile("hdfs://myfile")
val tups = queries.map(line => line.split('\t'))
val countHour = tups.map(x => (x(0) + "+" + x(2).take(13) +
val byUserHour = countHour.map(x => (x._1.split("\\+")(0),(
val byUser = byUserHour.groupByKey // RDD[(String, Iterable
val times = countHour.map(x => x._1.split("\\+")(1)).distin
val broadcastTimes = sc.broadcast(times)
```

To this

```
library(sparklyr)
library(dplyr)
sc <- spark_connect(master = "yarn-client")
queries <- spark_read_csv(sc = sc, name = "green",
                path = "hdfs://myfile")
```

sparklyr Functions

- `spark_` Connecting to Spark
- `spark_read_` and `spark_write_` Read and write Spark DataFrames from CSV, JSON and Parquet
- `sdf_` Operations on Spark DataFrames
- `ml_` Functions to invoke ML algos
- `ft_` Functions to transform Spark DataFrames
- Documentation

sparklyR vs. SparkR from Stack Overflow

sparklyr

- Does not support `do()` - arbitrary functions on groups or rows

SparkR (built into Spark 1.6+)

- More general front end to Spark using R

Installing sparklyr (well, other things too)

Use a Bootstrap Action

- Installs RStudio Server on Master node of EMR cluster
- Accessible via Web at ec2-ip-of-master-node.amazon.com:8787
- Login as hadoop user (with hadoop password)
- Sets all environment variables
- Also installs sparklyr, SparkR and other options

Caveats of this BA

- Older version of RStudio
- Specific to AWS EMR Hadoop configuration

Using sparklyr

Cluster

```r
# This assumes your environment variables are already set
library(sparklyr)
library(dplyr)

# To connect to a cluster
sc <- spark_connect(master = "yarn-client")
```

Local

```r
Sys.setenv(SPARK_HOME="/usr/local/Cellar/apache-spark/2.0.2
library(sparklyr)
library(dplyr)

sc <- spark_connect(master = "local")
```

Analyzing NYC Taxi Data (the new Big Data iris dataset)

Github Repo from RStudio Example

Setup

- Downloaded the NYC Taxi dataset used by (Todd Schneider)[`http://toddwschneider.com/posts/analyzing-1-1-billion-nyc-taxi-and-uber-trips-with-a-v`
and put it on s3 (s3://bigdatateaching)
- Created Parque files from the CSVs using sparklyr

What we'll see

- Read in Parquet files and create Spark DataFrames
- Do some simple munging and aggregations

Spark Parameters to tweak (YMMV)

- BA Used allocates all available resources to Spark
- spark.driver.cores
- spark.executor.cores
- spark.executor.memory

Additional Resources

- [http://www.agildata.com/ apache-spark-2-0-api-improvements-rdd-dataframe-datase
- [http://blog.revolutionanalytics.com/2016/10/ tutorial-scalable-r-on-spark.html]
- [https://www.toptal.com/spark/ introduction-to-apache-spark]
- [https://0x0fff.com/spark-architecture/]
- [https://0x0fff.com/spark-architecture-shuffle/]
- [https://0x0fff.com/spark-memory-management/]
- [https://0x0fff.com/apache-spark-future/]

Parting thoughts

- Like all Big Data tools, Spark is finicky and you should understand more about the internal workings of Spark. It is very powerful and fast when configured and used correctly
- I've been collecting large datasets that are publicly available but not necessarily readily available and/or usable on Amazon. Hosted on s3://bigdatateaching/
  - NYC Taxi Data (csv files and Parquet)
  - Criteo 1TB Dataset
  - More to come
- Also working on making the *ideal Data Science environment* setup easier using Ansible, which can be used on many platforms

Thank you!

- vaisman_marck@bah.com
- marck@datacommunitydc.org
- twitter.com/wahalulu