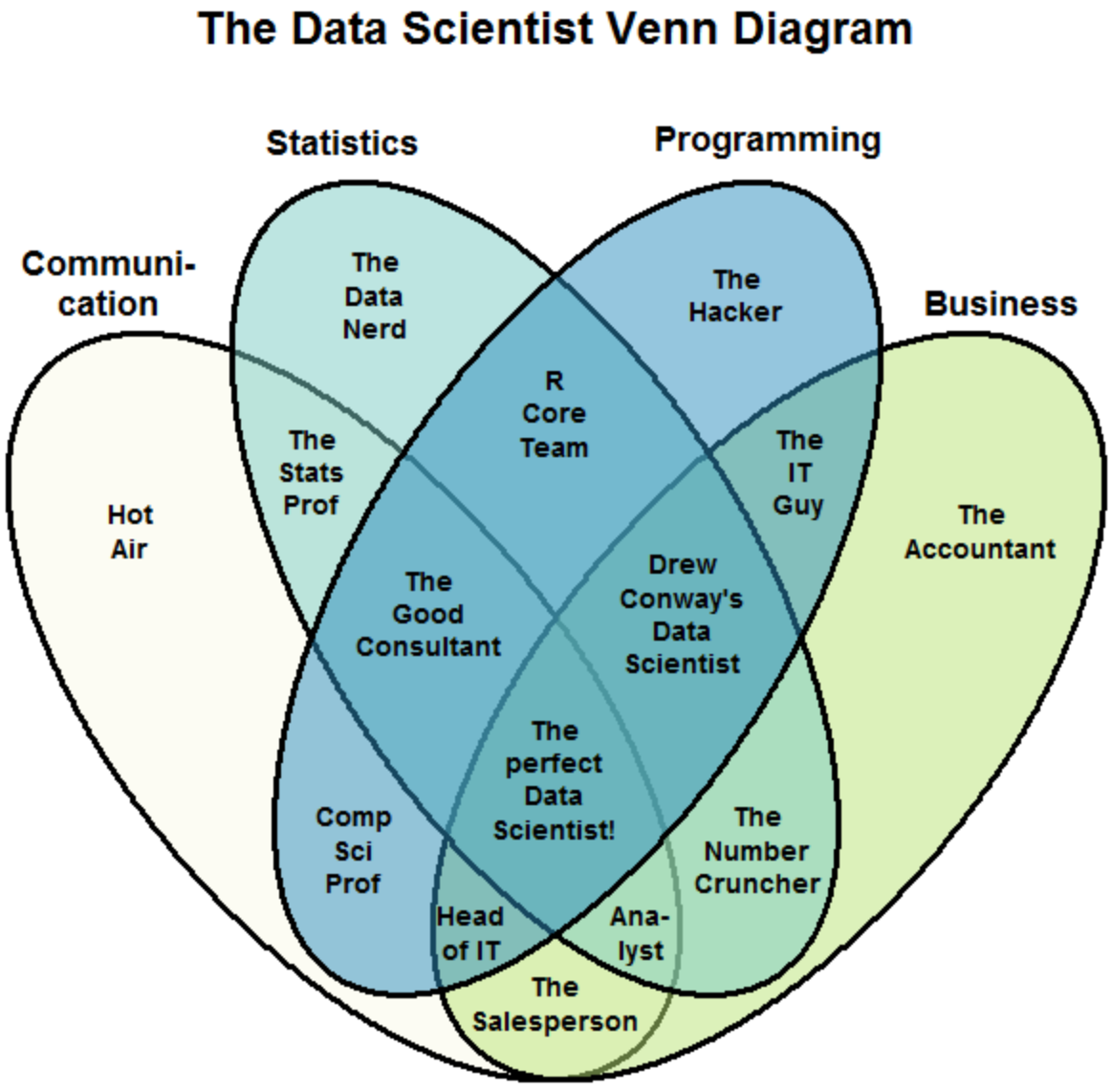
Scala for datascience

September 2016

Pradeep Gowda @ DataScience Indy



Courtesy of Stephan Kolassa on DataScience StackExchange.

Survey

- How many use Python
- How many use R
- How many use Scala+Python
- How many use Hadoop/Spark

Why Scala for DS

- Java ecosystem
- Better programming langauge
- Concurrency support
- Functional Programming
- Spark Ecosystem
- Strong open source community

Why Scala (contd...)

- Better programming language
  - Not Java
  - Funtional programming + OOPS
  - REPL
  - "Pythonish"

Why scala (contd... Functional Programming)

What is functional programming?

- Immutability
- Functions with no side effects
- Referential transparency
- Higher order functions

Why Scala (contd... oops)

- Object Oriented methods still useful
- ... and familiar
- classes
- inheritence
- .. methods ..
- blah...

How do Python compare?

- Python supports many of the OO Programming ideas
- and some FP (`map`, `reduce`, `apply`)
- `lambda` is severly restricted
- not really encouraged

How does R compare?

- R is actually more FP oriented than python
- In practice more procedural than FP
- Evaluating the design of R language; Morandat et. al from Purdue, 2012.

The balance between imperative and functional features is fascinating. We agree with the designers of R that a purely functional language whose main job is to manipulate massive numeric arrays is unlikely to be a success. It is simply too useful to be able to perform updates and have a guarantee that they are done in place rather than hope that a smart compiler will be able to optimize them. The current design is a compromise between the functional and the imperative; it allows local side effects, but enforces purity across function boundaries. It is unfortunate that this simple semantics is obscured by exceptions such as the super-assignment operator (<<−) which is used as a sneaky way to implement non-local side effects.

One of the most glaring shortcomings of R is its lack of concurrency support. Instead, there are only native libraries that provide behind-the-scenes parallel execution. Concurrency is not exposed to R programmers and always requires switching to native code. Adding concurrency would be best done after removing non-local side effects, and requires inventing a suitable concurrent programming model. One intriguing idea would be to push on lazy evaluation, which, as it stands, is too weak to be of much use outside of the base libraries, but could be strengthened to support parallel execution.

The object-oriented side of the language feels like an afterthought. The combination of mutable objects without references or cyclic structures is odd and cumbersome. The simplest object system provided by R is mostly used to provide printing methods for different data types. The more powerful object system is struggling to gain acceptance.

Why Scala (contd... java ecosystem)

- Instrumentation support
- Many man-centuries of experience around tuning the GC
- Cross platform
- Interoperability with Java
    - can use any Java class
    - can be called from Java (JVM lang)

Why Scala (contd... concurrency)

- High performance with Akka

Scala + Data science ecosystem

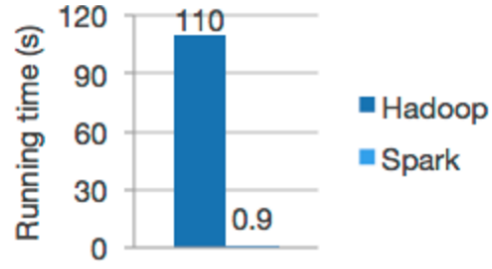
- Spark
- Cassandra
- Kafka

Apache Spark™ is a fast and general engine for large-scale data processing.

Speed

Run programs up to 100x faster than Hadoop MapReduce in memory, or 10x faster on disk.

Apache Spark has an advanced DAG execution engine that supports cyclic data flow and in-memory computing.



Logistic regression in Hadoop and Spark

Ease of Use

Write applications quickly in Java, Scala, Python, R.

Spark offers over 80 high-level operators that make it easy to build parallel apps. And you can use it *interactively* from the Scala, Python and R shells.

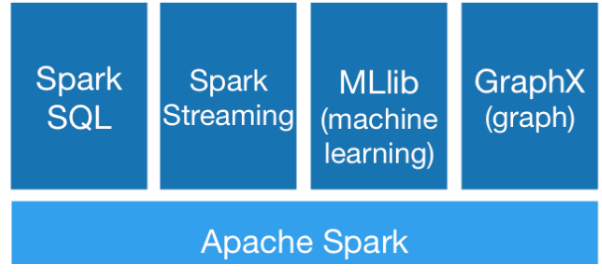
```
text_file = spark.textFile("hdfs://...")

text_file.flatMap(lambda line: line.split())
    .map(lambda word: (word, 1))
    .reduceByKey(lambda a, b: a+b)
```

Word count in Spark's Python API

Generality

Combine SQL, streaming, and complex analytics.

Spark powers a stack of libraries including SQL and DataFrames, MLlib for machine learning, GraphX, and Spark Streaming. You can combine these libraries seamlessly in the same application.



components of Spark

- SQL and dataframes
- MLLib for Machine Learning
- Spark Streaming

regression, isotonic regression,...
- Decision trees, random forests, and gradient-boosted trees
- Recommendation: alternating least squares (ALS)
- Clustering: K-means, Gaussian mixtures (GMMs),...
- Topic modeling: latent Dirichlet allocation (LDA)
- Feature transformations: standardization, normalization, hashing,...
- Model evaluation and hyper-parameter tuning
- ML Pipeline construction
- ML persistence: saving and loading models and Pipelines
- Survival analysis: accelerated failure time model
- Frequent itemset and sequential

Spark

- Spark is implemented in Scala
- Spark <-> python, a match made in ...
- Spark streaming (some of the features are not available for py API)
- MLLib leans heavily on parallel processing

Why not Scala

- Java ecosystem
- Scala the langauge can be scary

Scala is for you if ...

- You deal with large datasets
- You are already using Hadoop/Spark/Finagle
- You are a programmer/developer/software engr

You = the team you are part of, work with ...

Scala may not be for you if ...

(it's not you, it's me)

- You are more statistician than programmer
- You are more domain expert than programmer

Where to learn Scala

- EdX
- Programming in Scala by Odersky
- Programming Scala by Dean Wampler

Scala + DS

- Databricks community edition

Jupyter Notebook

- Scala kernel for Jupyter
- Zeppelin notebook
- IndyScala presentation