Introduction to PySpark

Andrew Crosby – Data Scientist @ AutoTrader

AutoTrader

Outline

1. Overview of Spark including standalone installation instructions
2. Example data analysis with data frames in both Pandas and Spark
3. Example of using Spark for machine learning

What is Apache Spark?

"Apache Spark™ is a unified analytics engine for large-scale data processing"

- https://spark.apache.org/

"Apache Spark is an open-source distributed general-purpose cluster computing framework with (mostly) in-memory data processing engine that can do ETL, analytics, machine learning and graph processing on large volumes of data at rest (batch processing) or in motion (streaming processing) with rich concise high-level APIs for the programming languages: Scala, Python, Java, R, and SQL."

- https://jaceklaskowski.gitbooks.io/mastering-apache-spark/spark-overview.html

"A parallelized version of Pandas and Scikit-Learn"- Me

Alternatives?

- Spark isn’t the only option for parallizing dataframes in Python

- Dask - https://docs.dask.org/en/latest/
  - Native Python solution
  - Pros and cons v Spark: http://docs.dask.org/en/latest/spark.html

How to install Spark in Standalone mode for local use

1. Install Python and Java; and add both to PATH. (For Java, both OpenJDK and OracleJDK should work.)
2. Download and extract the Spark binaries (http://spark.apache.org/downloads.html)
3. Set SPARK_HOME to point to the extracted directory
4. Install the findspark Python library (https://pypi.org/project/findspark/)

Windows specific steps

1. Use version 2.3.x of the Spark binaries and not version 2.4.x. There are issues with version 2.4.x on Windows: https://stackoverflow.com/questions/53252181/python-worker-failed-to-connect-back
2. Download winutils from https://github.com/steveloughran/winutils
3. Set HADOOP_HOME to point to “{extract_path}/winutils/Hadoop-2.7.1”

Test the installation

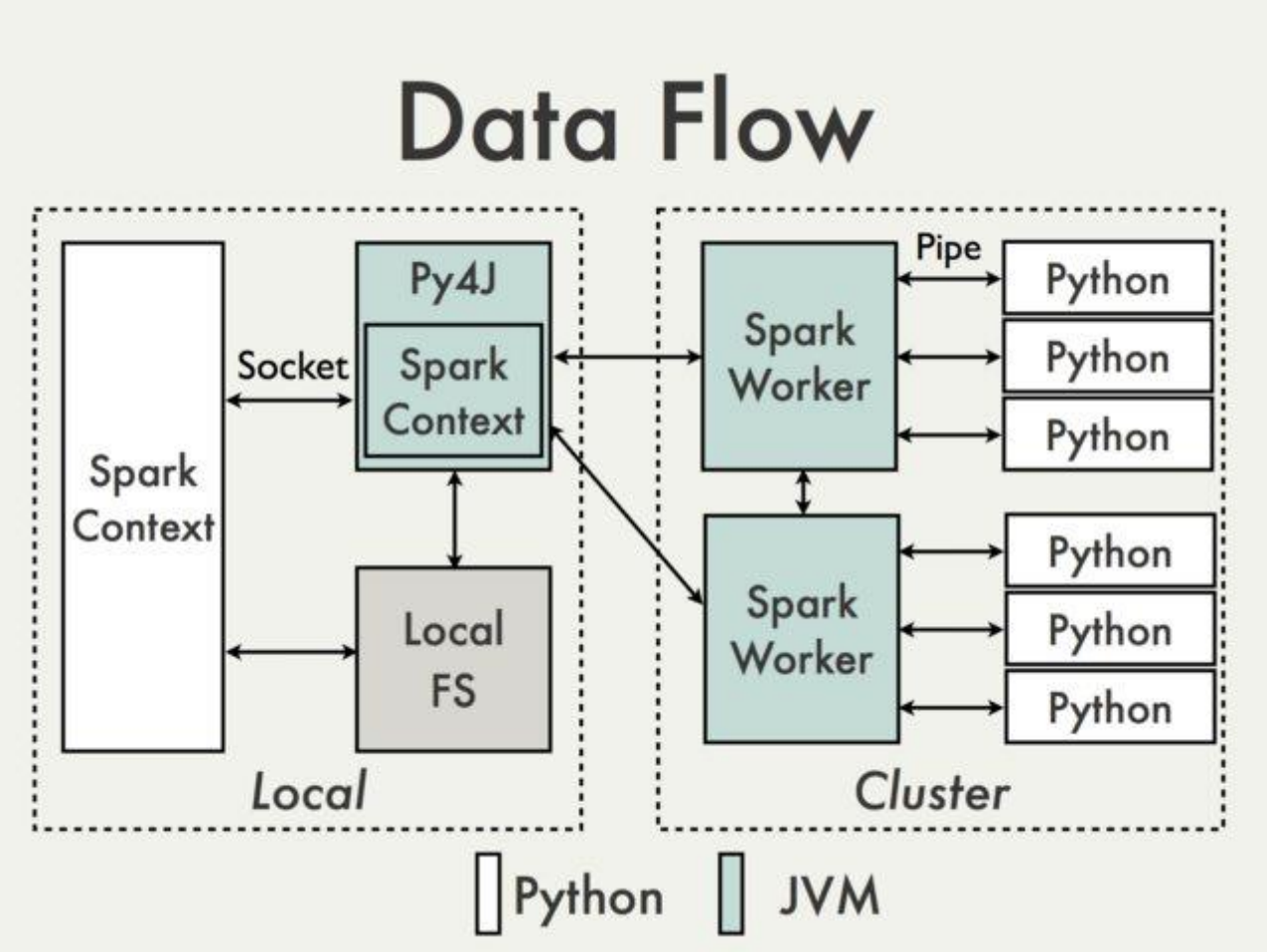
Jupyter notebook: 1 - Installation

Key features

- Distributed
- In memory
- Lazy query execution (transformations v actions)
- Predominantly written in Scala, but Python is very much a first class citizen
- Core low level data structure is the RDD (Resilient Distributed Dataset)
- Provides a high level DataFrame API
- Support for wide range of machine learning methods

Distributed

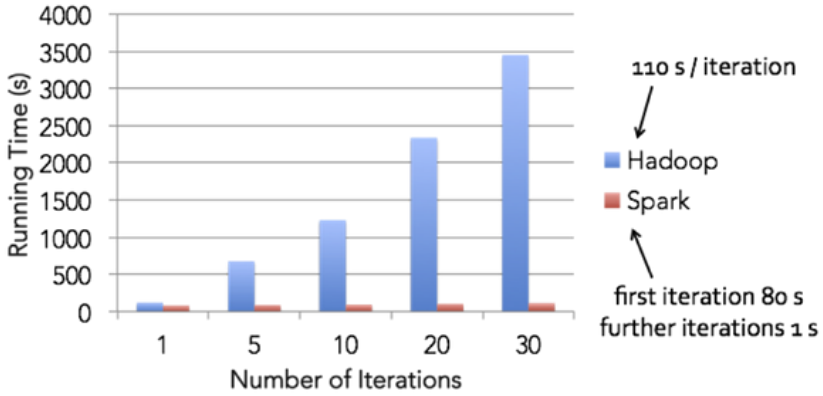
- One driver, many executors
- Driver:
  - Builds and optimizes query
  - Distributes tasks to executors
  - Collects results
  - Program using PySpark
  - Uses Py4J to launch JVM
- Executors:
  - Execute tasks
  - Run on JVM
  - Can launch Python to apply UDFs



https://cwiki.apache.org/confluence/display/SPARK/PySpark+Internals

In memory

- Evolution of MapReduce
- Much faster – reduced disk IO
- Makes iterative algorithms (e.g. ML) feasible



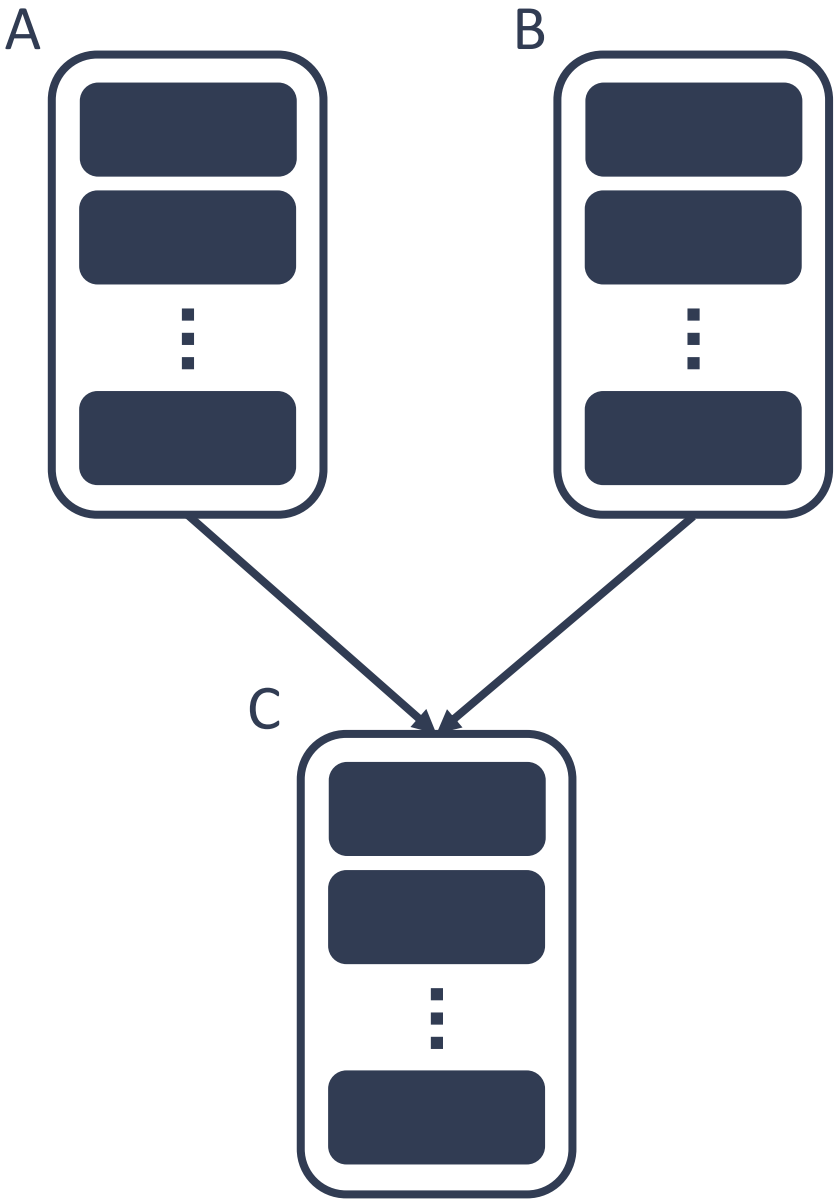
https://blog.cloudera.com/blog/2014/03/apache-spark-a-delight-for-developers/

Lazily evaluated queries

- Queries built up through transformations
  - `filter, join, groupBy, aggregate, dropDuplicates, limit, withColumn, …`
- Only executed when required by an action
  - `show, collect, write, toPandas, …`
- Allows query optimization via Spark's Catalyst optimizer
  - Predicate pushdown etc.

RDDs - Resilient Distributed Datasets

- **R**esilient – Spark stores the lineage of each dataset so knows how to recalculate it from the source data if necessary
- **D**istributed – Split across multiple partitions and possibly multiple machines
- **D**atasets – A collection of typed rows



Example revisited

Jupyter notebook: 2 - Example

Pandas example

Jupyter notebook: 3a - DataFrames - Pandas

PySpark equivalent

Jupyter notebook: 3b - DataFrames - PySpark

Machine Learning with PySpark

- Inspired by/evolution of pipelines in scikit-learn
- Transformers
  - Have a `transform` method
  - `DataFrame` -> `DataFrame`
  - `Tokenizer, StopWordsRemover, SQLTransformer, VectorAssembler, Interaction, …`
- Estimators
  - Have a `fit` method
  - `DataFrame` -> `Transformer`
  - `LinearRegression` (estimator) produces a `LinearRegressionModel` (transformer)
- Pipelines
  - Special estimator that chains together a series of transformers/estimators

Machine Learning with PySpark

Jupyter notebook: 4 - ML

Any questions?

AutoTrader