Python/Singularity Integration

Jon Schiefelbein

March 4, 2015

Abstract

This is a summary of using Python with Spark, ElasticSearch and Accumulo.

1 Python

All exploration, tests and code was written using Continuum Analytic's Anaconda Python distribution. This distribution was used as it has a simple installation, does not require administrative privileges and includes by default all the needed Python libraries. Python is rapidly becoming the lingua franca of data science. Python is one of only three languages able to interface with Apache Spark. Spark is written in Scala and as Scala is a JVM language Java is the other language. Python also has "mature" libraries for interfacing with ElasticSearch and Accmulo.

2 PySpark

PySpark, while initially a second class tool in versions of Spark prior to version 1, has become a viable, reliable and scalable tool for interacting with large data sets on a Hadoop cluster. My experiments with PySpark showed an initial learning curve as PySpark uses a functional programming style. Once the learning curve was overcome the resulting code was readable and compact with favorable performance on a 5 node Hadoop cluster. Furthermore the syntax between Python for Spark and Scala for Spark is similar enough even fairly novice Engineers or Data Scientists could easily translate from one to the other. This makes learning Spark easier as any documentation for Scala can be used to learn PySpark and vice versa.

No time was spent measuring performance differences between Spark and PySpark. However, PySpark translates all Python code to Java using a library called py4j. Given this I suspect performance differences would be marginal if they exist at all. Performance on data sets with tens to hundreds of thousands of points was acceptable. Data sets with millions of points was easily done in the 10 to 20 second range. It was when data sets with points exceeding 5 million was used that performance exceed what would be a reasonable time for web users to wait for a response.

3 ElasticSearch

Where Spark excels at data retrieval and calculations Elastisearch excels at retrieving specific sets of data. If you have a large data set, i.e. all of the weather observations for a 80 year period and wish to view only the observations for 1977, elasticsearch should be your tool of choice. There is still an initial learning curve for elasticsearch. One interesting note is it is easy to insert date into elasticsearch than to retrieve it. Once one learns how to make simple queries the ability to find specific points of data very quickly is impressive. The next phase of my research for elasticsearc is to retrieve results using elasticsearch and then use Spark to calculate results. Python is an ideal glue language for such a task and I suspect the integration of the two technologies will be particularly powerful.

Elasticsearch performance for retrieval of tens to tens of thousands of data points is very good. Like Spark performance on a 5 node cluster becomes an issue once more than 5 million data points are needed in bulk. Elasticsearch has streaming methods for this and I suspect the next stage of experiments will demonstrate an ability to retrieve data at speeds acceptable for use in web applications such as Singularity.

4 Accumulo

I have not explored using Accumulo with Python in any depth. Initial research shows using it with the pyaccumulo library will be another extremely power tool with possibly a lower learning curve than elasticseach with database like terminology of tables and rows. Combined with GeoWave's addition of geo-referencing and Python's ability to interact with all the other tools including arcpy this should be the next area of exploration.