using-spark-with-jupyter

December 24, 2018

1 Using Spark with Jupiter

This document is created as Jupyter Notebook and is available in different formats through export:
* Using Spark with Jupyter - HTML * Using Spark with Jupyter - Markdown * Using Spark with Jupyter - PDF * Using Spark with Jupyter - Jupyter Notebook .ipynb

1.1 Run Jupyter with Spark in Docker Container

Once you install docker, start a Jupyter container with spark.

```
> docker run -d --rm -p 18888:8888 -e GRANT_SUDO=yes -v C:\Sandbox\notebooks:/home/jovyan --nar
```

1.2 Install Spark with Jupyter on Windows

Following procedure helps setting up Spark with Jupyter notebook on Windows.

1. Instal Java - JDK 1.8+
2. Install Python 3
3. Install Spark
4. Download winutils.exe
5. Set Environment Variables
6. Verify Components
7. Start Jupyter Notebook

1.2.1 1. Install JDK 1.8

Go to Oracle's Java site to download and install latest JDK.
Will assume Java is installed at `C:\Program Files\Java\jdk1.8.0_191`

1.2.2 2. Install Python 3.7

Download and install Python from https://www.python.org/downloads/.
Will assume Python is installed at `C:\Python37`.

1.2.3 3. Install Spark

Download Spark from https://spark.apache.org/downloads.html:

- Spark release: 2.4.0 (Nov 02 2018)
- Package type: Pre-built for Apache Hadoop 2.7 and later
- Download: spark-2.4.0-bin-hadoop2.7.tgz

Extract Spark. Will assume Spark is extracted at: `C:\apps\spark-2.4.0-bin-hadoop2.7`
Modify log4j configuration to reduce log activity. Copy log4j.properties.template to log4j.properties. Modify:

```
log4j.rootCategory=WARN, console
```

1.2.4 4. Download winutils.exe

Download winutils.exe (http://media.sundog-soft.com/Udemy/winutils.exe or https://github.com/steveloughran/winutils) and place it under `%SPARK_HOME%\bin`.

1.2.5 5. Install Jupyter

```
pip install --upgrade pip
pip install jupyter
```

1.2.6 6. Set Environment Variables

```
set JAVA_HOME=C:\Program Files\Java\jdk1.8.0_191
set SPARK_HOME=C:\apps\spark-2.4.0-bin-hadoop2.7
set HADOOP_HOME=%SPARK_HOME%
set PATH=%PATH%;%SPARK_HOME%\bin
```

Open `Control Panel` and in the search box type `environment variables`. Click the `Edit the system environment variables` link.

- The System Properties dialog opens.
- Click the `Environment Variables...` button
- Add above variables. Suggest that you not use variable references, but specify the values fully. This way you avoid problems caused by unpredictable order of variable evaluation and assignment.

1.2.7 6. Verify Components

```
# Verify Python
> python --version
Python 3.7.1

# Verify Java
> java -version
java version "1.8.0_191"
Java(TM) SE Runtime Environment (build 1.8.0_191-b12)
Java HotSpot(TM) 64-Bit Server VM (build 25.191-b12, mixed mode)
```

Start `pyspark` and execute simple program

```
>>> spark.range(5).toDF("num").show()
+---+
|num|
+---+
|  0|
|  1|
|  2|
|  3|
|  4|
+---+
```

Press `Ctrl-Z` to exit.

1.2.8 7. Start Jupyter Notebook

Create a Spark bootstrap script and place it in `spark.py` file:
“‘python

```
In [19]: import os
         import sys
         import glob

         def get_spark(appName = 'HelloWorld'):
             spark_home = os.path.abspath(os.environ.get('SPARK_HOME', None))
             spark_python = os.path.abspath(spark_home + '/python')
             pyj4 = os.path.abspath(glob.glob(spark_python + '/lib/py4j*.zip')[0])
             if (not spark_python in sys.path):
                 sys.path.append(spark_python)
             if (not pyj4 in sys.path):
                 sys.path.append(pyj4)
             from pyspark.sql import SparkSession
             spark = SparkSession.builder.appName(appName).getOrCreate()
             return spark

         spark = get_spark()
```

Navigate to your notebook directory (`C:\Sandbox\notebook`) and start Jupyter:

```
jupyter notebook
```

Create a new Python 3 notebook and execute following into a cell:

```
%run spark.py
```

Inspect the `spark` variable:

```
In [21]: spark
```

```
Out[21]: <pyspark.sql.session.SparkSession at 0x281af449e80>
```

In a new cell you can run simple Spark program:

```
In [23]: spark.range(5).toDF('num').show()
```

```
+---+
|num|
+---+
|  0|
|  1|
|  2|
|  3|
|  4|
+---+
```

1.3 Use `findspark` Package

`findspark` package provides `findspark.init()` function to make pyspark importable as a regular library.

For more information on the package, see the findspark github page.

First install the findspark package.

```
pip install findspark
```

```
In [1]: import findspark
```

By default `findspark` uses the `SPARK_HOME` environment variable. To override this behavior, specify spark home directory:

```
findspark.init('/path/to/spark')
```

```
In [2]: # Use SPARK_HOME
        findspark.init()
```

```
In [15]: # Check where Spark is found
         findspark.find()
```

```
Out[15]: 'C:\\apps\\spark-2.4.0-bin-hadoop2.7'
```

Now Spark packages can be accessed using `import`.

```
In [3]: from pyspark.sql import SparkSession
```

```
In [6]: spark = SparkSession.builder.master('local[4]').appName("Hello World").getOrCreate()

        # Inspect SparkSession

        spark
```

```
Out[6]: <pyspark.sql.session.SparkSession at 0x281af449e80>

In [16]: # Inspect SparkContext, associated with the session
         spark.sparkContext

Out[16]: <SparkContext master=local[4] appName=Hello World>

In [17]: # Get configuration for the SparkContext
         spark.sparkContext.getConf().getAll()

Out[17]: [('spark.app.name', 'Hello World'),
          ('spark.master', 'local[4]'),
          ('spark.rdd.compress', 'True'),
          ('spark.serializer.objectStreamReset', '100'),
          ('spark.driver.port', '60942'),
          ('spark.executor.id', 'driver'),
          ('spark.submit.deployMode', 'client'),
          ('spark.driver.host', 'LYOGA'),
          ('spark.ui.showConsoleProgress', 'true'),
          ('spark.app.id', 'local-1545656532472')]

In [18]: # What is available as methods and attributes for SparkContext
         dir(spark.sparkContext)

Out[18]: ['PACKAGE_EXTENSIONS',
          '__class__',
          '__delattr__',
          '__dict__',
          '__dir__',
          '__doc__',
          '__enter__',
          '__eq__',
          '__exit__',
          '__format__',
          '__ge__',
          '__getattribute__',
          '__getnewargs__',
          '__gt__',
          '__hash__',
          '__init__',
          '__init_subclass__',
          '__le__',
          '__lt__',
          '__module__',
          '__ne__',
          '__new__',
          '__reduce__',
          '__reduce_ex__',
          '__repr__',
```

```
'__setattr__',
'__sizeof__',
'__str__',
'__subclasshook__',
'__weakref__',
'_accumulatorServer',
'_active_spark_context',
'_batchSize',
'_callsite',
'_checkpointFile',
'_conf',
'_dictToJavaMap',
'_do_init',
'_encryption_enabled',
'_ensure_initialized',
'_gateway',
'_getJavaStorageLevel',
'_initialize_context',
'_javaAccumulator',
'_jsc',
'_jvm',
'_lock',
'_next_accum_id',
'_pickled_broadcast_vars',
'_python_includes',
'_repr_html_',
'_serialize_to_jvm',
'_temp_dir',
'_unbatched_serializer',
'accumulator',
'addFile',
'addPyFile',
'appName',
'applicationId',
'binaryFiles',
'binaryRecords',
'broadcast',
'cancelAllJobs',
'cancelJobGroup',
'defaultMinPartitions',
'defaultParallelism',
'dump_profiles',
'emptyRDD',
'environment',
'getConf',
'getLocalProperty',
'getOrCreate',
'hadoopFile',
```

```
 'hadoopRDD',
 'master',
 'newAPIHadoopFile',
 'newAPIHadoopRDD',
 'parallelize',
 'pickleFile',
 'profiler_collector',
 'pythonExec',
 'pythonVer',
 'range',
 'runJob',
 'sequenceFile',
 'serializer',
 'setCheckpointDir',
 'setJobDescription',
 'setJobGroup',
 'setLocalProperty',
 'setLogLevel',
 'setSystemProperty',
 'show_profiles',
 'sparkHome',
 'sparkUser',
 'startTime',
 'statusTracker',
 'stop',
 'textFile',
 'uiWebUrl',
 'union',
 'version',
 'wholeTextFiles']
```

In [ ]: