PySpark: How to install and Integrate with the Jupyter Notebook

29 OCT 2015 in tutorials

At Dataquest, we've released an interactive course on Spark, with a focus on PySpark. We explore the fundamentals of Map-Reduce and how to utilize PySpark to clean, transform, and munge data. In this post, we'll dive into how to install PySpark locally on your own computer and how to integrate it into the Jupyter Notebbok workflow.

Some familarity with the command line will be necessary to complete the installation.

Overview

At a high level, these are the steps to install PySpark and integrate it with Jupyter notebook:

1. Install the required packages below
2. Download and build Spark
3. Set your enviroment variables
4. Create an Jupyter profile for PySpark

Required packages

- Java SE Development Kit
- Scala Build Tool
- Spark 1.5.1 (at the time of writing)
- Python 2.6 or higher (we prefer to use Python 3.4+)
- Jupyter Notebook

Java

Spark requires Java 7+, which you can download from Oracle's website:

- Mac download link
- Linux download link

Spark

Head to the Spark downloads page, keep the default options in steps 1 to 3, and download a zipped version (.tgz file) of Spark from the link in step 4. Once you've downloaded Spark, we recommend unzipping the folder and moving the unzipped folder to your home directory.

Scala build tool

1. To build Spark, you'll need the Scala build tool, which you can install:

- Mac: `brew install sbt`
- Linux: instructions

1. Navigate to the directory you unzipped Spark to and run `sbt assembly` within that directory (this should take a while!).

Test

To test that Spark was built properly, run the following command in the same folder (where Spark resides):

```
bin/pyspark
```

and the interactive PySpark shell should start up. This is the interactive PySpark shell, similar to Jupyter, but if you run `sc` in the shell, you'll see the SparkContext object already initialized. You can write and run commands interactively in this shell just like you can with Jupyter.

Environment variables

Environment variables are global variables that any program on your computer can access and contain specific settings and pieces of information that you want all programs to have access to. In our case, we need to specify the location of Spark and add some special arguments which we reference later.

Use `nano` or `vim` to open `~/.bash_profile` and add the following lines at the end:

```
export SPARK_HOME="$HOME/spark-1.5.1"
export PYSPARK_SUBMIT_ARGS="--master local[2]"
```

Replace `"$HOME/spark-1.5.1"` with the location of the folder you unzipped Spark to (and also make sure the version numbers match!).

Jupyter profile

The last step is to create a profile for Jupyter specifically for PySpark with some custom settings. To create this profile, run:

```
Jupyter profile create pyspark
```

Use `nano` or `vim` to create the following Python script at the following location:

```
~/.jupyter/profile_pyspark/startup/00-pyspark-setup.py
```

and then add the following to it:

```
import os
import sys

spark_home = os.environ.get('SPARK_HOME', None)
sys.path.insert(0, spark_home + "/python")
sys.path.insert(0, os.path.join(spark_home, 'python/lib/py4j-0.8.2.1-s

filename = os.path.join(spark_home, 'python/pyspark/shell.py')
exec(compile(open(filename, "rb").read(), filename, 'exec'))

spark_release_file = spark_home + "/RELEASE"

if os.path.exists(spark_release_file) and "Spark 1.5" in open(spark_re
    pyspark_submit_args = os.environ.get("PYSPARK_SUBMIT_ARGS", "")
```

```
    if not "pyspark-shell" in pyspark_submit_args:
        pyspark_submit_args += " pyspark-shell"
        os.environ["PYSPARK_SUBMIT_ARGS"] = pyspark_submit_args
```

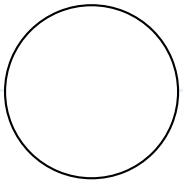
If you're using a later version than Spark 1.5, replace "Spark 1.5" with the version you're using, in the script.

Run

To start Jupyter Notebook with the `pyspark` profile, run:

```
jupyter notebook --profile=pyspark
```

To test that PySpark was loaded properly, create a new notebook and run `sc` in one of the code cells to make sure the SparkContext object was initialized properly.



Srini Kadamati

Data scientist at Dataquest.io (Learn Data Science in your Browser). Loves tacos and BBQ. Texan living in California. Get in touch @srinikadamati.

5 Comments Dataquest 1 Login

Recommend Share Sort by Oldest

Join the discussion…

Egor Ignatenkov · a month ago

jupyter notebook says: "Unrecognized alias: '--profile=pyspark', it will probably have no effect."

Reply · Share ›

> **Vik Paruchuri** Mod → Egor Ignatenkov · a month ago
>
> This may be due to a newer version of Jupyter no longer supporting the flag. I think Srini used IPython 3.
>
> Reply · Share ›

essay writers online · 13 days ago

Machine learning is very complicated to learn. It involves different codes that contain complex terms. All the data's that you have posted was relevant to a common data structures that are applicable in the common data frame.

Reply · Share ›

Samantha Zeitlin · 13 days ago

It would help if you could show an example of what the expected output should look like if the spark context was initialized properly?

Reply · Share ›

Pushpam Choudhury · 4 days ago

How can I add some user-defined property such as username, in the SparkConf object? I can hardcode the details in pyspark.context by using setExecutorEnv, but how for security I would like to use the details(username etc) captured from the notebook's login page. Can you please suggest a viable approach?