SPIP: Pure Python Package in PyPI (Spark Connect)

Author: Hyukjin Kwon
JIRA: SPARK-47540
Prototype: apache/spark#45053

Q1. What are you trying to do? Articulate your objectives using absolutely no jargon.

As part of the Spark Connect development, we have introduced Scala and Python clients. While the Scala client is already provided as a separate library and is available in Maven, the Python client is not. This proposal aims for end users to install the pure Python package for Spark Connect by using `pip` install `pyspark-connect`.

The pure Python package contains only Python source code without jars, which reduces the size of the package significantly and widens the use cases of PySpark. See also Introducing Spark Connect - The Power of Apache Spark, Everywhere'.

Q2. What problem is this proposal NOT designed to solve?

This proposal does not aim to

- Change existing PySpark package, e.g., `pip install pyspark` is not affected
- Implement full compatibility with classic PySpark, e.g., implementing RDD API
- Address how to launch Spark Connect server. Spark Connect server is launched by users themselves
- Local mode. Without launching Spark Connect server, users cannot use this package.
- Official release channel is not affected but only PyPI.

Q3. How is it done today, and what are the limits of current practice?

Currently, we run `pip install pyspark`, and it is over 300MB because of dependent jars. In addition, PySpark requires you to set up other environments such as JDK installation.

- This is not suitable when the running environment and resource is limited such as edge devices such as smart home devices.
- Requiring a non-Python environment is not Python friendly.

Q4. What is new in your approach and why do you think it will be successful?

It provides a pure Python library, which eliminates other environment requirements such as JDK, and reduces the resource usage by decoupling Spark Driver, and reduces the package size.

Q5. Who cares? If you are successful, what difference will it make?

Users who want to leverage Spark in the limited environment, and want to decouple running JVM with Spark Driver to run Spark as a Service. They can simply pip install pyspark-connect that does not require other dependencies (except Python dependencies just like other Python libraries).

Q6. What are the risks?

Because we do not change the existing PySpark package, I do not see any major risk in classic PySpark itself. We will reuse the same Python source, and therefore we should make sure no Py4J is used, and no JVM access is made. This requirement might confuse the developers. At the very least, we should add the dedicated CI to make sure the pure Python package works.

Q7. How long will it take?

I expect around one month including CI set up. In fact, the prototype is ready so I expect this to be done sooner.

Q8. What are the mid-term and final “exams” to check for success?

The mid-term goal is to set up a scheduled CI job that builds the pure Python library, and runs all the tests against them.

The final goral would be to properly test end-to-end usecase from pip installation.