Introduction to Spark Connect

Spark v3.4

Simona Scala

University of Bologna

July 2023

Outline

Apache Spark Overview

What is Spark?

Apache Spark is a powerful and versatile distributed computing framework designed to process and analyze large-scale data with high efficiency and speed. It provides an in-memory computing engine, enabling real-time data processing and iterative data analysis, making it ideal for big data analytics and machine learning applications.

Key Features of Apache Spark

- High-level APIs in Java, Scala, Python, and R
- Spark SQL for SQL and structured data processing
- Pandas API for pandas workloads
- MLlib for machine learning tasks
- GraphX for graph processing

Resource Management and Cluster Integration

At its core, Spark manages resources through a robust resource management system, which is crucial for optimizing task execution across a cluster of computing nodes.

Resource management in Spark involves efficiently utilizing the available resources, such as CPU cores, memory, and storage, to execute tasks across the cluster. The cluster manager monitors the resources and ensures that they are distributed and allocated effectively among multiple Spark applications running simultaneously.

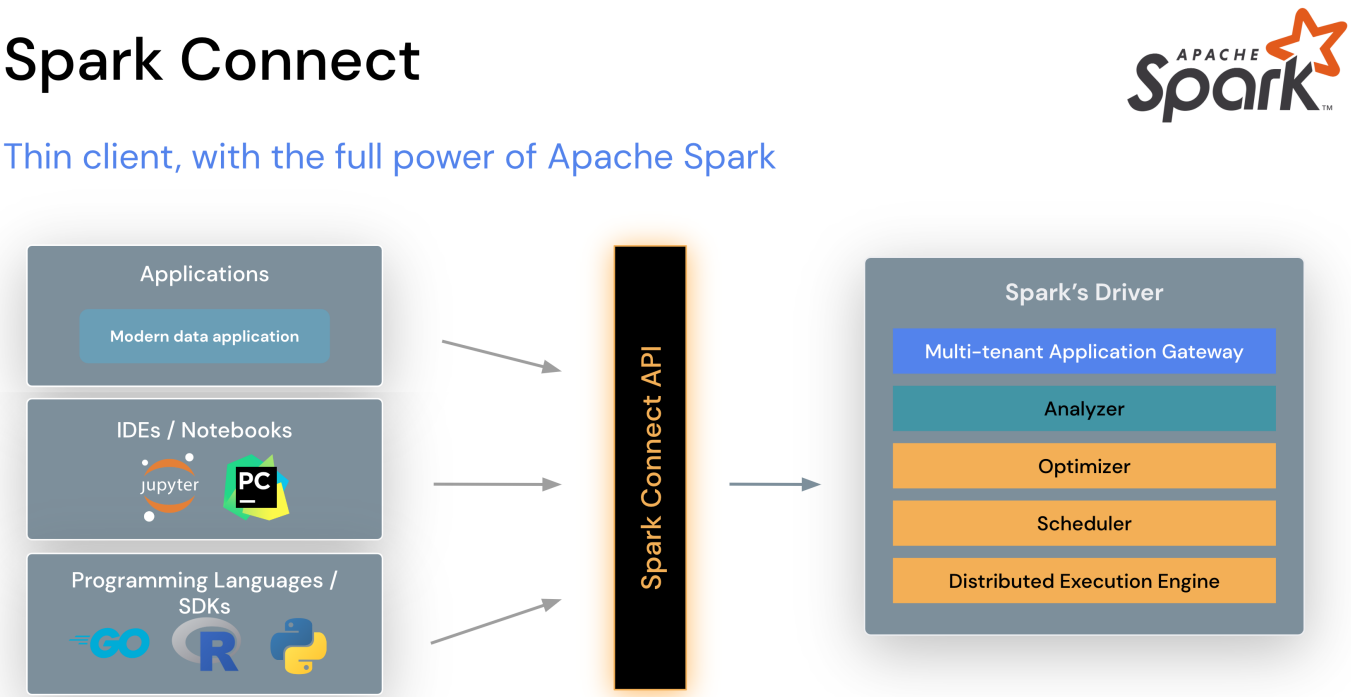
Spark can rely on various cluster managers to operate in different cluster environments (e.g. Apache Hadoop YARN).

Spark Connect

Remote Access to Spark Clusters

In Spark 3.4, a new feature called "Spark Connect" was introduced. It allows Spark client applications to connect with Spark clusters remotely. By this way, users can leverage Spark and its open ecosystem from anywhere and to integrate it into different applications. It also brings DataFrame API support for PySpark and DataFrame/Dataset API support in Scala, making it easier to work with data on Spark clusters.

Simplified Spark Application Development



The Spark Connect client library is a tool that makes it easier to develop Spark applications. It's a simple and lightweight API that can be used in various places, like application servers, IDEs, notebooks, and different programming languages.

How Does Spark Connect API Work

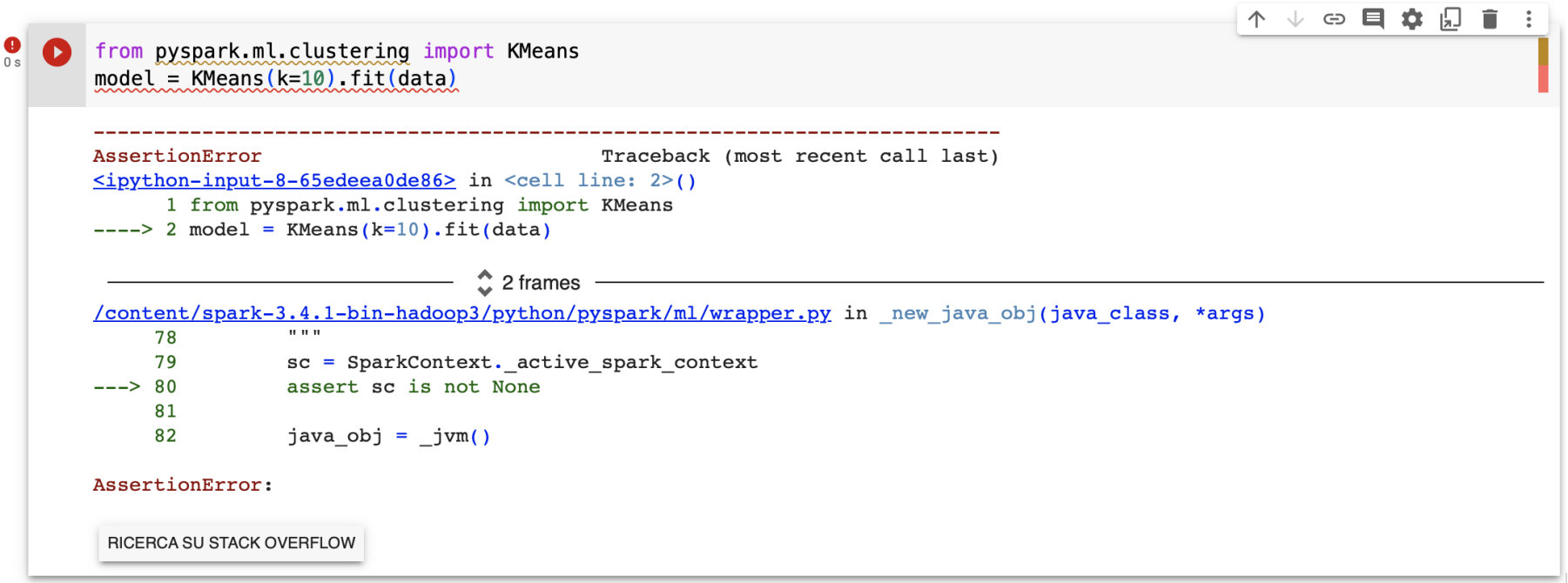
1. The Spark Connect client turns DataFrame operations into "unresolved logical query plans." These plans are like step-by-step instructions for Spark, but they don't depend on any specific programming language
2. The client sends unresolved logical query plans to the server using a framework called "gRPC"
3. The Spark Connect endpoint on the Spark server receives the instructions and translates them into Spark's logical plan operators
4. The standard Spark execution process kicks in, ensuring efficient execution on the Spark cluster
5. The results are then sent back to the client using gRPC, in a format called Apache Arrow-encoded row batches

Benefits Of Spark Connect

Spark Connect provides operational benefits that address multi-tenant issues, which arise when multiple users or applications share the same computing resources. With this new architecture, Spark Connect mitigates these challenges by offering

- Stability: Applications run in their own processes, minimizing the impact of resource-intensive applications on others and avoiding conflicts with the Spark driver
- Upgradability: The Spark driver can be upgraded independently, allowing applications to benefit from improvements and security fixes without disruption
- Debuggability and Observability: Spark Connect enables interactive debugging and monitoring of applications using native metrics and logging libraries, simplifying issue identification and resolution in multi-tenant environments

What is supported in Spark 3.4



```
from pyspark.ml.clustering import KMeans
model = KMeans(k=10).fit(data)
```

```
---------------------------------------------------------------------------
AssertionError                            Traceback (most recent call last)
<ipython-input-8-65edeea0de86> in <cell line: 2>()
      1 from pyspark.ml.clustering import KMeans
----> 2 model = KMeans(k=10).fit(data)

 ──────────────────────────────────────  2 frames  ──────────────────────────────────────
/content/spark-3.4.1-bin-hadoop3/python/pyspark/ml/wrapper.py in _new_java_obj(java_class, *args)
     78             """
     79             sc = SparkContext._active_spark_context
---> 80             assert sc is not None
     81
     82             java_obj = _jvm()

AssertionError:
```

RICERCA SU STACK OVERFLOW

In Spark 3.4, Spark Connect supports most PySpark APIs, including DataFrame, Functions, and Column. However, some APIs such as SparkContext and RDD are not supported. You can check which APIs are currently supported in the API reference documentation. Supported APIs are labeled "Supports Spark Connect" so you can check whether the APIs you are using are available before migrating existing code to Spark Connect.

Use Spark Connect in a Google Colab Notebook

Method 1: Manual Installation

1. Download Java Virtual Machine (JVM)
2. Download the latest version of Apache Spark
3. Set up the environment variables for Spark
4. Install and import the library for locating Spark
5. Test the installation by starting a "traditional" Spark session
6. Now that the Spark server is running, connect to it remotely using Spark Connect

Follow this notebook for detailed steps and code.

Method 2: Automatic Installation

1. Use `pip install` to install PySpark
2. Create a remote Spark session

Follow this notebook for detailed steps and code.