**Temi Moses**

**Databricks vs Apache Spark**

**Summary**

Databricks and Spark seem to be buzzwords in the data community. Over the last few years these technologies/softwares have advanced to be one of the most popular in the Data Engineering world. Both created by the same people to solve different problems, yet both so different in its approach and philosophy. This paper analyzes the functionalities, pros and cons of these two tools, and how they interact with one another.

**Databricks**

Databricks is a cloud-based data engineering tool used for processing and transforming massive quantities of data and exploring the data through machine learning models. Databricks is a data platform that lets you run data analytics projects in a structured manner with the help of open source technologies, which allows for faster speeds and scalability. Some of the open source technologies include Machine Learning libraries, Tensorflow and visualization tools, as well as some quality of life additions.

The following are Databricks integrations that currently offered:

| **Databricks** | **Integrations** |
| --- | --- |
| TIBC Spotfire ☑️  Tableau ☑️  Cassandra ☑️  Pentaho ☑️  Redis ☑️  Amazon Kinesis ☑️ | Amazon Redshift ☑️  Talend ☑️  Alteryx ☑️  MongoDB ☑️  Looker ☑️ |

A key benefit of Databricks is its ability to facilitate the streamlining of the solutions for problems and its design for the utility of Apache Spark. Databricks Cloud connectivity enables Databricks to compress data from the unified Spark engine which uses higher-level libraries and supports data streaming, graph processing, SQL queries, and machine learning. It mitigates the complexity in handling the processes and permits easy adaptability for the productivity of developers. As a cloud-native platform, it encourages collaboration among stakeholders who may be working with different programming languages such as Python, Scala, R, and SQL for machine learning models. Another characteristic is the interactivity of the platform makes it easier for the users to visualize by point and click and scripting options like D3, ggplot, and matplotlib. The multilevel data security of Databricks is another critical benefit that is inherent in the platform, which has options for adjusting access for identity management, fine-grained, auditing, compliance standards, and data encryption.

According to Microsoft Azure below are some of Databricks key functionalities:

### **Optimized spark engine**

Simple data processing on autoscaling infrastructure, powered by highly optimized Apache Spark™ for up to 50x performance gains.

**Machine learning run time**

One-click access to preconfigured machine learning environments for augmented machine learning with state-of-the-art and popular frameworks such as PyTorch, TensorFlow, and scikit-learn.

**MLflow**

Track and share experiments, reproduce runs, and manage models collaboratively from a central repository.

**Choice of language**

Use your preferred language, including Python, Scala, R, Spark SQL and .Net—whether you use serverless or provisioned compute resources.

**Collaborative notebooks**

Quickly access and explore data, find and share new insights, and build models collaboratively with the languages and tools of your choice.

### **Delta lake**

Bring data reliability and scalability to your existing data lake with an open source transactional storage layer designed for the full data lifecycle.

### **Native integrations with Azure services**

Complete your end-to-end analytics and machine learning solution with deep integration with Azure services such as Azure Data Factory, Azure Data Lake Storage, Azure Machine Learning, and Power BI.

**Interactive workspaces**

Enable seamless collaboration between data scientists, data engineers, and business analysts.

(Microsoft)

**Cons of Databricks**

Databricks lacks in having a better form of localized testing, when it was primarily OSS Spark; it was easier to test/manage releases versus the newer Databricks Runtime. Databricks also lacks in graphing support as compared to its counterparts.

**Spark**

According to Apache Spark’s website “Apache Spark is a data processing framework that can quickly perform processing tasks on very large data sets, and can also distribute data processing tasks across multiple computers, either on its own or in tandem with other distributed computing tools.” (Spark) Apache Spark application consists of two main components: a driver, which converts the user's code into multiple tasks that can be distributed across worker nodes, and executors, which run on those nodes and execute the tasks assigned to them. Some form of cluster manager is necessary to mediate between the two. Spark gains a speed advantage over previous technologies of a similar type by storing results of intermediate operations in memory.

**Features**

A Spark application runs as independent processes, coordinated by the SparkSession object in the driver program.The resource or cluster manager assigns tasks to workers, one task per partition. A task applies its unit of work to the dataset in its partition and outputs a new partition dataset. Because iterative algorithms apply operations repeatedly to data, they benefit from caching datasets across iterations. Results are sent back to the driver application or can be saved to disk.

Spark is a quick and standard processing engine compatible with Hadoop data. It can compute with Hadoop clusters via Spark’s standalone mode or YARN. Also it can process data in Cassandra, Hive, Hadoop InputFormat, and HBase. It is built to perform interactive queries, machine learning, batch processing and new workloads.

**Con**

Spark’s limitations include its deficiency in memory management. Also its PySpark functionality is not as robust as scala. Other disadvantages include no support for real-time processing, challenges with small files, it is not economical, and it lacks a dedicated file management system. It also tends to have higher latency when compared to its counterparts. Hence, these are the reasons why users could potentially switch to Spark’s competitor.

**How they interact**

Some years ago Databricks was fundamentally an easy-to-deploy and maintain platform for running Apache Spark. Azure Databricks has the most current versions of Apache Spark and permits a smooth integration with open source libraries. Apache Spark environment manages the Spin up clusters with the global scale and availability of Azure. Clusters are built to ensure reliability and performance without the constraint for a need to monitor.

**Conclusion**

Although created by the same people, Databricks and Apache Spark have a lot of differences. Databricks integrates with more systems, has more capabilities, and is a more user friendly platform. Things like external ML frameworks and Data Lake connection management make Databricks a more powerful analytics engine than its counterparts Apache Spark. Overall Databricks is becoming the more popular alternative to Spark, and it’s popularity is only growing with the years.

**Citation**

​​Azure Databricks. (n.d.). Retrieved from https://azure.microsoft.com/en-us/services/databricks/#features

Databricks Review: Pricing, Pros, Cons & Features. (2019, October 16). Retrieved from https://comparecamp.com/databricks-review-pricing-pros-cons-features/

Kusnetzky, D. (2014, July 01). Databricks makes Hadoop and Apache Spark easy to use. Retrieved from https://www.zdnet.com/article/databricks-makes-hadoop-and-apache-spark-easy-to-use/

Mahmood, O. (2021, June 07). What does Databricks do? Retrieved from https://towardsdatascience.com/what-does-databricks-do-8a6c4ef9071b

Soh J., Singh P. (2020) Apache Spark, Big Data, and Azure Databricks. In: Data Science Solutions on Azure. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-6405-8\_6