VS

Databricks

PREPARED BY

Saturn Cloud

When data scientists are choosing a data science platform, they seek a platform that is easy to use, that scales with their team, and that allows them to use their preferred tools and languages.

Saturn Cloud and Databricks are two popular platforms among data scientists. They both allow data scientists to do their work in the cloud using hosted notebooks, but they differ significantly in their approach and features.

Saturn Cloud

Saturn Cloud is a platform that allows data scientists and teams to work with scalable resources in the cloud. With just a few clicks, Saturn Cloud provides access to computing resources with customizable amounts of memory and power, including GPUs and Dask distributed computing clusters, in a completely hosted environment. In Saturn Cloud, data scientists can use their preferred languages, IDEs, and machine learning libraries. It offers full Git integration, shared custom images, and secure credential storage as well, making it easy to scale and build any data science team in the cloud. And with features like jobs and deployments, it supports the entire machine learning lifecycle from experimentation to production. With its intuitive user interface and enhanced configurability, data science in the cloud becomes effortless and work becomes more productive.

Databricks

Databricks has been a leading tool that allows data scientists to perform computations, analytics, and model training across large data sets by using clusters of computers. Databricks is a storage solution, but it features an analysis layer where data scientists can employ an interactive notebook to complete analysis. The notebooks simultaneously support SQL, Scala, Python, and R. The center offering of Databricks is access to Spark clusters, allowing users to scale their data science analysis in the cloud. With Databricks, users also have access to several other components, such as a storage layer, a feature store, and a model registry. In this sense, Databricks supports the end-to-end data science workflow from storage to production. However, it lacks several key features that data science teams look for, such as flexibility and interoperability.

Choosing Your Data Science Platform

As you prepare to choose a data science platform for your team, consider how the following features align with your business priorities and needs:

Interactive Data Science Environment

Data scientists can do their work better in an environment that is predictable and easy to integrate.

With Saturn Cloud, data scientists primarily interact with the resources through a hosted JupyterLab or RStudio. They can also interact with the resources through SSH if they want to use their preferred IDE. The interface lets data scientists run Python and R scripts and notebooks, as well as any terminal commands. In addition, Saturn Cloud is flexible enough that Dask clusters can be used from a local machine, bypassing the worker entirely.

By using JupyterLab and RStudio, the Saturn Cloud environment gives data scientists the exact user experience that they typically get when opening a notebook in their local environment. Because Saturn Cloud uses popular tools, it integrates into data scientists' existing processes without requiring any changes to their code architecture or methods. Any code that works on a data scientist's local computer should run the exact same way in the Saturn Cloud workspace.

Meanwhile, Databricks defaults to a custom interactive notebook environment that simultaneously supports SQL, Scala, Python, and R but that comes with limitations inherent to any custom solution. Notably, data scientists must use Databricks libraries (e.g., dbutils), which may or may not work in their local environment. What's more, with Databricks, data scientists can't necessarily import custom modules to their notebook environment. To circumvent this problem, data scientists need to use plug-ins to employ other environments and IDEs.

These limitations undermine best practices in software development, especially in terms of code modularity and reuse, and they make unit testing difficult. With Databricks, data scientists can't replicate the experience they have on their local computer, and they will likely need to create a new process to write code and translate it for the Databricks environment.

Open and Interoperable Computing

Users should be able to choose their own tools and workflows in a data science platform.

Libraries

With Saturn Cloud, data scientists can install libraries and modules to their environment at a resource level using pip, apt, or conda or via a start script. Or they can create predefined packages of libraries and modules called images that have the required packages. These images have almost no constraints on what can be installed and what versions of different packages can be used. Further, if desired, the images can be based on Docker containers that a data science team already has available, further increasing their flexibility. So, environment setup code that exists outside of Saturn Cloud can be immediately put into an image and reused by whatever projects need it.

In Databricks, data scientists typically install libraries via the workplace library system. This user interface allows them to select from a number of possible library types (e.g., Python Egg, Python Wheel files), as well as some options from other languages (e.g., CRAN for R). If that framework doesn't work for a particular library, data scientists can install the library by writing an init script that runs at cluster startup. In addition, data scientists can modify the Docker containers that are used for the Spark cluster, but here, they will experience constraints on how the containers are formatted and located.

Data scientists frequently encounter dependency problems, like when a script hasn't been formatted into a full library or like when two different libraries are required but they have conflicting dependencies. Because of the many different approaches to installing dependencies in Databricks, data scientists will struggle to track which libraries will show up at runtime and how dependencies will change over time. In contrast, data scientists using Saturn Cloud will be better able to manage their environments, which break less often and require less debugging.

Git Repositories

With Saturn Cloud, users have the ability to easily use the full functionality of git within a resource. Each resource can be connected to a repository at which time users can push, pull, and do other commands. A resource isn't limited to a single git repository. Instead, resources can contain as many git repositories as users want.

Although Databricks allows users to connect a notebook to a git repository, it uses a custom methodology: instead of users having full git control, they are limited to committing to a single branch of a git repository. This functionality is simple and accommodates many typical data science tasks, but it does not meet data scientists' needs for more complex behaviors.

Scalable Performance

Scalable computing is central to any data science platform.

Instance Types

Saturn Cloud only uses AWS. Databricks allows users to utilize AWS, Azure, and Google Cloud backends for computation.

Regarding AWS, for the most part, both Saturn Cloud and Databricks offer identical instance types. Databricks does offer slightly more options for small- and medium-sized instances, but it notably maxes out at 768 GB of RAM and 96 vCPUs per instance for large-sized instances. In comparison, Saturn Cloud's upper range is 4 TB and 128 vCPUs, so users can run bigger workloads that require instances of these sizes.

Dask vs. Spark

Scaling code is a profoundly powerful tool for data science. It enables data scientists to take existing code and make it parallelized and distributed. Scaling code results in dramatically faster work and the possibility to use larger data than ever before.
In Saturn Cloud, data scientists can employ easy-to-use Dask clusters to scale their code. Dask–an open-source distributed data processing platform–is written in Python, so users must access it from other programs written in Python. Dask can be run locally on a single machine, or it can be distributed over a group of them. Dask uses a scheduler to coordinate work (supporting frameworks such as Kubernetes and YARN, among many others). Dask workers communicate with each other, and the Dask scheduler passes information between the workers and the client process.

In Databricks, users can employ Spark clusters to scale their code. Like Dask, Apache Spark is a large-scale and open source data processing platform. It is written in Scala and thus runs in the Java Virtual Machine. However, users can connect to Spark and run commands from other languages (e.g., Python, R) by using packages written specifically for those languages (e.g., PySpark, sparklyr). Spark can either be run as a standalone scheduler, or it can be connected to systems like Kubernetes and YARN.

Dask presents several advantages to data scientists, particularly to those who work in Python:

- Dask can be easier to debug than Spark, because Dask is written in Python and doesn't require an interface layer. In contrast, Spark requires data scientists to be able to debug Java, and its interface layer could possibly introduce additional points of failure.

- Dask provides a more universal system for running Python code over a cluster, giving users more control. With Dask, users can easily execute Python code with arbitrary data requirements, including no data whatsoever. Although Spark does have the capability to execute Python code within each cluster node, it is designed around its RDD system, which is written with the MapReduce paradigm of a single dataset. This single dataset is evenly split up, and the same computation is applied to each split. But many tasks–like those requiring complex calls such as recursion and external data loading–don't align with this design.

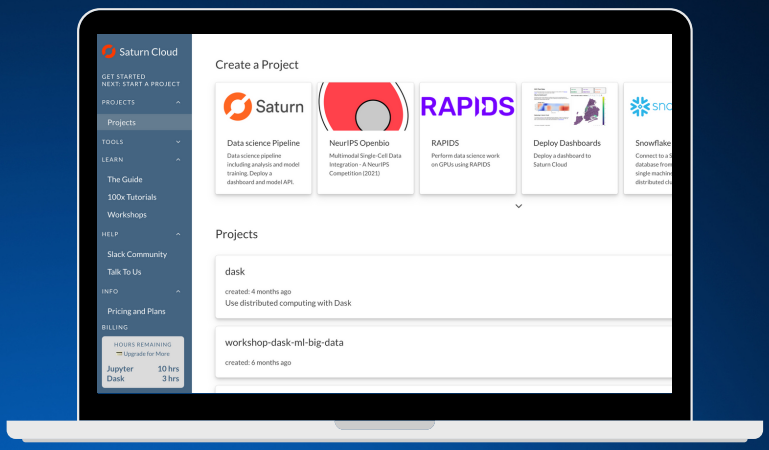
- Dask can perform more quickly than Spark under many conditions. For example, with RAPIDS and GPUs, Dask performs 2,000 times more quickly than Spark. With Dask, data scientists will see that it takes less time to run code from Python and that the code itself is faster.

Importantly, Saturn Cloud does not require Dask. If data scientists using Saturn Cloud do not want to use Dask, they don't need to. They can run a single node server to complete their computations. In contrast, Databricks does require Spark. If data scientists using Databricks want to run a non-Spark workflow, they would still need to spin up a cluster that has the Spark driver and workers. Although they could run a single node cluster, they would still need to run the Spark driver on the single node. With Saturn Cloud, data scientists have more flexibility to choose their workflow, only scaling when doing so is required.

Ready to give it a try? Join Saturn Cloud for free and get 30 hours of free compute each month.

Saturn Cloud

Your open data science cloud environment



ALL-IN-ONE WORKSPACE

- Scalable data science with Python, R, & more
- Use up to 4TB RAM, large GPUs, and Dask clusters
- Cloud-hosted JupyterLab and RStudio
- Easily share work with colleagues
- Straightforward deployment environments
- Run jobs plus deploy dashboards and APIs
- Connect from existing cloud resources
- Admin reporting tools

Trusted by

NVIDIA. snowflake® VITAL PROTEINS® Cellarity checkout.com Senseye

And more

Join thousands of data scientists today and get 30 hours of free compute each month:

JOIN FOR FREE

www.saturncloud.io