Available online www.jsaer.com

***Journal of Scientific and Engineering Research*, 2021, 8(12):253-263**

Research Article

ISSN: 2394-2630
CODEN(USA): JSERBR

Real-world Use Cases of Databricks in Big Data Projects

Ravi Shankar Koppula

Email: Ravikoppula100@gmail.com

Abstract As data becomes a critical asset for businesses, the need for efficient and scalable data processing frameworks has never been greater. Apache Spark, a powerful big data technology, addresses these demands with its user-friendly interface and rapid performance. Databricks, a managed service built on Apache Spark, further enhances this capability by offering a cloud-based platform that streamlines the development and deployment of data products. This paper explores several real-world use cases of Databricks in large-scale data projects, highlighting its role in data ingestion, transformation, exploration, analysis, and machine learning. Databricks accelerates AI development and promotes secure innovation through collaborative data science and engineering workflows. This unified data analytics platform supports businesses in leveraging big data for predictive modeling, reporting, and real-time analytics, ultimately facilitating scalable and efficient data management.

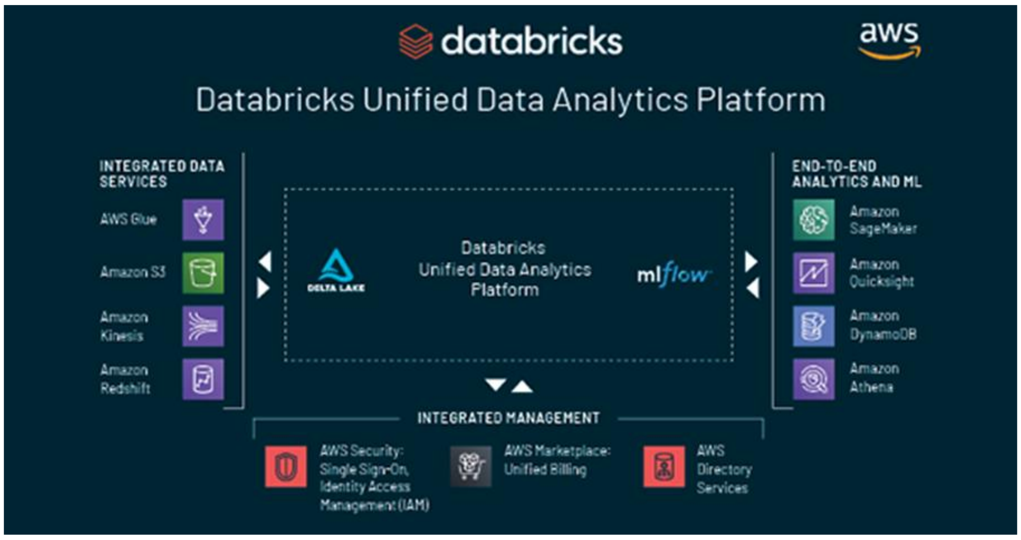
Keywords Databricks, Apache Spark, Big Data Analytics, Data Processing, Machine Learning, Data Engineering, Cloud Computing, Data Lakehouse, Delta Lake, ML flow, Data Management, Data Integration, Predictive Analytics, Real-time Data Processing, Scalable Data Solutions

1. Introduction

Major corporations are now generating a growing array of data products as data becomes increasingly valuable. These products, which range from predictive models to traditional reporting tools, necessitate advanced data pipelines for their development, deployment, and upkeep. Although big data technologies such as Apache Hadoop and Spark enable these pipelines, they also introduce new challenges. Data engineers and scientists must now possess proficiency in intricate technologies and dedicate a substantial amount of time to managing the infrastructure required for data-intensive enterprise projects. Apache Spark is a widely-used big data technology renowned for its user-friendly interface, rapid performance, and modular architecture. Databricks, a startup established by Spark's creators, expands the technology to provide a cloud-based, managed service. Databricks integrates an optimized version of the Apache Spark platform with an intuitive, collaborative web-based interface, with the goal of enabling big data users to concentrate on creating data products rather than managing large-scale data infrastructure. This paper examines real-world instances of Databricks in big data projects.

1.1. Overview of Databricks as a Unified Data Analytics Platform

This article examines several real-world scenarios in which Databricks has been utilized in large-scale data projects, as well as the integration, automation, and orchestration of Databricks with the CData Data Integration Platform to expedite AI development. It showcases the use of Databricks in data processing, analytics, AI, and ML, while detailing the CData Data Integration platform, which allows organizations of all sizes to utilize CData Data Connectors for connecting Databricks with data from diverse sources. These Data Connectors simplify the use of SQL with Databricks for querying, transforming, and processing data, and are designed for real-time capabilities. By using CData Connectors, Databricks users can effectively work with live data, establishing collaborative AI structures that promote secure innovation, as well as an easily scalable, automated data pipeline between Databricks and various data sources. Databricks is an integrated data analytics platform created by the developers of Apache Spark. It is a cloud-based platform hosted on Azure that facilitates easy and collaborative development. Users have the ability to install libraries and packages for machine learning and deep learning, as well as develop AI applications with large-scale data. In addition, Databricks Delta is offered for data engineering, providing a managed environment for efficient data processing. This platform unifies data engineering, data science, and business processes within an open-source collaborative platform that promotes innovation, securely integrated with the Microsoft Azure Cloud. Databricks accelerates the AI lifecycle with its easily scalable, automated, and collaborative cloud-based platform.[1]


databricks
aws
Databricks Unified Data Analytics Platform
INTEGRATED DATA SERVICES
AWS Glue
Amazon S3
Amazon Kinesis
Amazon Redshift
DELTA LAKE
Databricks Unified Data Analytics Platform
mlflow
END-TO-END ANALYTICS AND ML
Amazon SageMaker
Amazon Quicksight
Amazon DynamoDB
Amazon Athena
INTEGRATED MANAGEMENT
AWS Security: Single Sign-On, Identity Access Management (IAM)
AWS Marketplace: Unified Billing
AWS Directory Services

2. Databricks in Data Ingestion

In this article, we will delve into two popular methods of data intake using Databricks. The first involves utilizing the built-in tools and magic commands within the platform's user interface. The second approach involves writing code in Python, making use of the capabilities of the Spark ecosystem. This framework is applicable to all programming languages and must be tailored accordingly. We will present various real-world scenarios that demonstrate the practical use of Databricks in handling big data. To begin with, we will focus on data ingestion. Databricks has been likened to a high-performance sports car - elegant and powerful, yet somewhat challenging to fasten seat belts in. This is because Databricks, as a higher-level abstraction layer, conceals the intricate details, enabling developers to achieve high levels of productivity with Spark (the underlying big data engine). Data ingestion serves as the seat belt for any significant big data project, playing a crucial role in maintaining data integrity and security. With Databricks, users may need to forego some of the convenience of the UI's clicks and magic commands in exchange for the capabilities of coding and the inherent integrations offered by Spark.

Databricks streamlines large-scale data management with its comprehensive collaborative platform that integrates big data and data science. The Unified Data Analytics Platform enables businesses to expedite creativity by consolidating data science, engineering, and business functions. Established by the original architects of Apache Spark, Databricks offers an open and cohesive platform for supporting joint development.[14]

2.1. Real-time Data Ingestion with Databricks

For real-time data ingestion without the Databricks Connectors API, you will have a few XML descriptors to hand-code the ingest schema, which looks something like this. There's nothing particularly complicated or convenient about the proprietary Connector XML, you just add your custom ingest code. This is an example in Scala. For streaming ingest (values might change in the update), write a function that appends a DataFrame to the Delta Tables dataframe and use Spark ForeachBatch, as in the example below. With batch ingest, you can write an appending DataFrame with an SQL principal table which also handles the update. For complex data types or structures, you should consider performing the write using a specialized operation utilizing Delta's MERGE INTO statement.

For Amazon S3, the recommended best practice for real-time data ingestion is to use AWS Lambda with the Simple Storage Service (S3). For Azure Blob Storage, set up an event-driven pipeline from blob storage to your Databricks cluster. With GCP, connect Cloud Storage to PubSub with Cloud Functions to trigger push subscriptions to PubSub.

Databricks simplifies real-time big data analytics by combining Apache Spark with a 1st-tier persistent data service, Delta, in a single, auto-scaling cluster. But, Databricks isn't opinionated about how data should get into that cluster. It has connectors for all the major messaging and big data platforms, but these just load data at rest. Modern architecture now separates the real-time ingest custom code (which should run in the ACID Spark cluster) from the messaging system using scalable compute like AWS Lambda. This solution is both cheaper and more resilient.[9]

3. Databricks for Data Transformation

The integration of data pipelines and automated cluster management in Databricks streamlines the large-scale data transformation process. Data engineers can effortlessly create data pipelines for ETL tasks with a simple click, as Databricks takes care of all cluster management and scheduling. This frees up data engineers to concentrate on constructing the pipelines, rather than dealing with infrastructure and scheduling. Furthermore, Databricks offers an enhanced version of Apache Spark that simplifies schema management, making it easy to convert and standardize intricate data types, thereby minimizing the potential for errors associated with manipulating nested fields with different configurations.[7]

Data transformation involves converting data from its original structure to a more suitable format for a specific purpose. This is a crucial aspect of managing data for analytics, business intelligence, and data science. The Databricks Unified Data Analytics Platform streamlines data transformation by bringing together data engineering, data science, and business analytics tasks. With this platform, you can utilize tools such as Apache Spark to extract, transform, and load extensive datasets from different sources, and then process the data using SQL, structured streaming, and machine learning libraries.

3.1. ETL Processes with Databricks

The primary advantages of using Databricks for the ETL process include faster extract, transform, and load processing, simplified batch processing of pipelines, and thorough error-handling that integrates with Apache Spark Structured Streaming for real-time data processing in the ETL pipeline. The Databricks environment also provides monitoring and automated logging for code development using MLflow, as well as improved monitoring of batch ETL processing through advanced job scheduling for Apache Spark jobs coordinated by the Databricks Job Scheduler. Consequently, Databricks offers a structured, cloud-native approach for data engineering, data science, and processing of big data and real-time data.

Databricks speeds up and simplifies the ETL process by creating an organized system for bringing together data engineering, collaborative data science, and handling big data and real-time data in the cloud. The Databricks platform combines its Unified Databricks Runtime with the Unified Data Service, which includes the Databricks Delta and Databricks Connect APIs. The Databricks Delta streamlines data pipeline and batch processing, while Databricks Connect enables interaction between the Databricks environment and the customer's IDEs.

4. Databricks for Data Exploration and Analysis

Databricks provides a user-friendly, scalable, and collaborative environment for data exploration. It combines the power of Apache Spark for distributed processing with familiar data analysis tools and libraries, making it a valuable platform for data scientists and analysts of all experience levels.

Databricks offers several features that make it suitable for data exploration:

Ease of Use:

- **Built-in Libraries:** Databricks runtime comes pre-installed with popular data exploration libraries like pandas, NumPy, and Spark SQL, allowing you to start exploring data immediately without installing additional packages.

- **Interactive Notebooks:** Databricks utilizes notebooks as the primary development environment. These notebooks allow you to write code, visualize data, and document your findings in a single, interactive environment. This makes data exploration an iterative process where you can test hypotheses and refine your analysis as you go.

Data Access:

- **Integration with Cloud Storage:** Databricks integrates seamlessly with various cloud storage platforms like AWS S3, Azure Blob Storage, and Google Cloud Storage. This allows you to explore data directly from your cloud data lake without needing to download it locally.
- **Apache Spark:** Databricks leverages Apache Spark for large-scale data processing. Spark enables you to explore massive datasets efficiently by distributing the workload across a cluster of machines.

Data Manipulation and Analysis:

- **Spark SQL:** Spark SQL provides a familiar SQL-like interface for querying structured data. This allows data analysts and SQL programmers to leverage their existing skillset for data exploration within Databricks.
- **DataFrames and Datasets:** Databricks offers DataFrames and Datasets, which are distributed in-memory representations of your data. These data structures facilitate efficient manipulation and analysis of large datasets.
- **Visualization Libraries:** Databricks notebooks support popular data visualization libraries like Matplotlib, Seaborn, and Plotly. These libraries allow you to create insightful visualizations to explore trends, patterns, and relationships within your data.

Collaboration:

- **Version Control:** Databricks notebooks are version controlled, allowing teams to collaborate on data exploration tasks and track changes made to the analysis.
- **Sharing and Publishing:** Notebooks can be easily shared with other users or published for wider access within an organization. This facilitates knowledge sharing and fosters collaboration among data scientists and analysts

4.1. Interactive Data Exploration with Databricks

Databricks is ideal for data exploration because you can easily switch between running small snippets of code and running complete, complex production jobs. This means that any logic you develop to explore your data at the snippet level can be easily applied right up to the job level without requiring further implementation. Databricks supports languages like SQL, Python, R, and Scala. Here, we will generally use SQL because it’s the most efficient way to manipulate data when using Spark. To view a table within the Databricks environment, run the following command. This will output the first 1000 rows of the specified table. You can also add a WHERE clause to this command. There is no real reason to use any other limit because if a table is too large, you will likely need to apply specific filters to view the data correctly.

Data exploration is a key part of any data project and is always performed as a first step. With traditional big data tools, this was hard to achieve. Usually, data presented at each stage underwent a series of complex processing and was transformed into a different format so it could be queried at a reasonable speed. Tools like Presto, Impala, or Redshift help to address this issue, but Databricks, used in combination with Spark SQL, is the most seamless way for such interactive data exploration.

5. Databricks for Machine Learning

MLflow is an open-source project by Databricks that aims to be the industry standard for machine learning lifecycle. It has four components: MLflow Tracking, MLflow Projects, MLflow Models, and MLflow Model Registry. It has been well adopted by the machine learning community, and Databricks is using it internally as well as contributing back new features and improvements. The Tracking Server implemented in MLflow is capable of logging the parameters, metrics, and model artifacts from different Machine Learning frameworks and tools. It enables users to easily manage their models across different running environments. With the feature of search and filter, users can quickly locate the desired model. But the current deployment of the MLflow Tracking Server lacks the horizontal scalability needed for enterprise usage. Taking advantage of the API provided by MLflow, users can write their own customized server with horizontal scalability as a microservice. In this way, the API server can delegate read and write requests to different independent storages for different flavors. With the help of the Proxy design pattern, Composition pattern, and Factory design pattern, users can achieve the goal of horizontal scalability for the API server with low coupling and high cohesion.

One major area of big data problems is the handling of large-scale machine learning. Model training often requires big iterative computation that can easily exceed the capacity of a single node. Databricks developed a highly scalable Machine Learning Library (MLlib) on top of Spark to enable scalable machine learning. MLlib complements users' distinct workloads and existing ecosystem by providing high-level APIs that can be easily integrated with users' custom code and data pipeline. With MLflow, Databricks provides an open-source platform to manage the machine learning lifecycle that is seamless with MLflow integration. And many organizations are using MLflow along with custom frameworks to effectively manage and automate model workflows.

5.1. Building ML Models with Databricks

In summary, by providing a unified approach to both big data processing and machine learning, Databricks proposes to reduce the complexities associated with machine learning in production environments. The Jupyter notebooks and Apache Spark clusters accelerate the model building processes. The collaborative environment and clusters can help reduce the time from model development to deployment.

End-to-End Pipelines: Databricks allows for the easy building of end-to-end pipelines, from ingesting data to visualizing the model performance.

Combined Infrastructure: Machine learning can often have a separate infrastructure. Databricks proposes that by combining it with the big data processing infrastructure, it can reduce the complexity of deploying ML models.

Collaboration: Data scientists can work together and share best practices. They can collaborate with engineers and scientists who are developing the production data pipelines. Databricks has a unified approach and collaborative environment across big data processing and machine learning. It proposes that the way you build machine learning models should be alongside and integrated with the rest of the big data pipeline. In this way, it provides a few developments in how machine learning pipelines are architected:

6. Databricks for Real-time Data Processing

In conclusion, Databricks is a powerful, cloud-native platform that simplifies the architectures of big data, real-time, and machine learning pipelines. Its direct integration with the major cloud vendors (AWS, Azure, and GCP) makes it a flexible choice for organizations that are already committed to a cloud provider. The scalability and performance of Spark, combined with the simplicity of the Databricks platform, reduce the development time of the pipelines while increasing the productivity of data engineers, data scientists, and developers.

In many organizations, big data and machine learning pipelines are still implemented using separate technologies, tools, and processes. Using a unified platform like Databricks can help consolidate some of the specialized tools in each pipeline. The integration of Databricks with Deep Learning libraries like Keras and TensorFlow make it a great tool for developing Deep Learning applications. The Databricks Runtime ML also provides the performance optimizations required to make Deep Learning feasible on large datasets. Databricks File System (DBFS) is another unique feature of the platform that simplifies the architecture of big data pipelines compared to external blob storage and HDFS.

Real-time data processing is the use case that Databricks is most frequently associated with. Databricks' structured streaming APIs build on the Spark SQL and DataFrames API, making the process of building, running, and deploying real-time big data applications much simpler compared to traditional, low-level streaming APIs such as those provided by Apache Flink and Apache Storm. Complex event processing is another use case for real-time data processing, and Databricks' scalable, cloud-native platform makes using it in conjunction with Spark very beneficial. The ability to mix batch and streaming workloads in the same codebase is a unique feature of Databricks that increases developer productivity. The Databricks Connect feature allows data engineers and data scientists who prefer to use local IDEs to easily work with the Databricks platform. The Databricks CLI provides similar benefits for users who prefer to work with command-line interfaces.

6.1. Stream Processing with Databricks

Stream processing with Apache Spark is efficient in Databricks because Databricks unifies the data engineering and data science workflows, and Databricks' runtime is built on top of Spark, which allows for faster development cycles and better performance. Databricks' interactive workspace is cloud-hosted and collaborative, making fast exploratory development easy. Databricks' optimized Spark runtime overcomes many Spark performance challenges, such as latency issues, checkpoint management, and lost and inconsistent data handling, allowing for scalable and reliable near real-time processing. Stream processing in Databricks is often done with the Databricks' Unified Data Analytics runtime, which adds continuous processing capabilities to Databricks, making this powerful platform even more versatile. With Databricks Delta and MLflow, stream processing in Databricks can easily support production-ready real-time machine learning. All of these features together enable fast, scalable, and reliable near real-time stream processing in Databricks.

Databricks simplifies big data and streamlines an organization's analytics strategy. From real-time stream processing and machine learning to production reporting, Databricks is a powerful platform that can do it all. Stream processing can be especially challenging to implement in a scalable manner across an organization, but Databricks makes it easier than ever. In this paper, you will learn how stream processing works and how you can do it with Databricks, as well as some real-world use cases.[4]

7. Databricks for Data Visualization

However, the above script executes on the driver node to plot the figure in Matplotlib. If you have a large dataset that is distributed and you would like to plot a figure using Matplotlib, this will not work well. You should use Databricks' display function to create figures and leverage both the power of Matplotlib and Databricks' distributed computing.

The above command will create a simple line plot and the output would be something as shown below:

~~~
# Plot
plt.plot(x, y)
plt.xlabel('X axis')
plt.ylabel('Y axis')
plt.title('Simple line plot created using Matplotlib')
plt.show()
```
# Data
x = [1, 2, 3, 4]
y = [10, 15, 13, 18]
```python
import matplotlib.pyplot as plt
~~~

Matplotlib is a comprehensive library for creating static, animated, and interactive plots that are consistently rendered across platforms using an API that closely resembles Matlab. Below is the code to create a simple line plot using Matplotlib in Databricks.

Let's look at one such example using Matplotlib.

Visualizing data is one of the most powerful methods to convey results, and it helps stakeholders grasp the essence of the underlying messages present in the data. Databricks leverages popular libraries such as Matplotlib, Seaborn, ggplot, Plotly, and Bokeh to create stunning visualizations.

7.1. Visualizing Data with Databricks

Databricks comes with built-in visualization capabilities. You can display tables or query results as pie charts, bar charts, line charts, or scatter plots using the display function of Databricks. In this chapter, we will demonstrate how to use Databricks to visualize cryptocurrencies and their historical performance. To follow along with these examples, you will need to set up a free Databricks Community account. Databricks provides an easy way to interact with big data. With its integrated work environment, users can write SQL, Python, R, and Scala code against their data. This code can also be easily shared and scheduled using the Databricks platform. Visualizing big data is an important topic. Traditional charts tend to be slow when plotting large

volumes of data. With Databricks, users can take full advantage of the big data processing capabilities. For example, you can summarize the data in a series of bar plot counts and then plot the data using the Databricks display function in just a few seconds.
Databricks provides an easy way to interact with big data. With its integrated work environment, users can write SQL, Python, R, and Scala code against their data. This code can also be easily shared and scheduled using the Databricks platform. Visualizing big data is an important topic. Traditional charts tend to be slow when plotting large volumes of data. With Databricks, users can take full advantage of the big data processing capabilities. For example, you can summarize the data in a series of bar plot counts and then plot the data using the Databricks display function in just a few seconds.

8. Databricks for Data Governance and Security

Often, ACLs and column-level restrictions are implemented in standard data warehouses to control access to data. When moving to the world of big data, the problem space is a little different. Cloud-based storage means that data can be stored very cheaply, but querying that data from services such as Spark is much more expensive (in terms of both time and money). One approach is to use services such as AWS's Glue to construct a Data Catalog. This can then be used by Databricks to manage table access metadata and use the Metastore Secrets to safely store the table access information. This has the advantage of avoiding hard-coded credentials and supporting fine-grained role-based access control (RBAC) for data in Delta Lake.[11]
One of the oft-overlooked elements of big data processing, especially with the promise of cloud computing, is actually ensuring that the right data is made available in the right way so that users can make the most of the data that is available to them. Data governance is essential to make sure that data is repeatedly and accurately served in a way that makes it most valuable and informative to users accessing that data. For some use cases, data access patterns need to be considered during the ETL process to permission data appropriately and reduce the risk of certain types of security breaches. Next, we look at how Databricks can make data governance and security easier.[5]

8.1. Ensuring Data Security with Databricks

Access to data hosted in the Databricks Unified Data Service (UDS) is secured through management of access controls for any of the integrated data sources (e.g., AWS S3, Azure Data Lake, Azure Blob Storage, GCP Storage, ADLS Gen2, JDBC, Delta Lake). Outsiders are prevented from accessing tables, views, or data stored in the Databricks Workspace with a cluster running. When jobs and clusters are run, the secret scopes command that is executed to access the secret values is dbutils. This command reads and writes secret values to a secret scope, ensuring runtime secrets are securely accessed during the job runs. Data and connections, structured or unstructured, must be securely implemented and managed with the highest possible security layers in the Databricks platform to ensure protection from data breaches and loss of sensitive data, while adhering to compliance rules and regulations like the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act (HIPAA). The discussion is supported by a healthcare fraud detection implementation example using the Databricks platform.
Security is an important aspect in any big data project implementation. Databricks has been successful in ensuring the data security, reliability, ease of use, and performance of the platform. In this case study, we provide practical real-world Databricks use cases to build, deploy, and scale big data, AI, and machine learning solutions.

9. Databricks for Cost Optimization in Big Data Projects

When cost or budget is a major concern in Big Data projects, Databricks has the answer. The unified platform can greatly help reduce overall costs in many ways. First of all, the architecture of the Databricks platform is such that the run-time costs of the analytical code can be lowered adequately. By using the serverless optimized approach, Databricks runtime executes code in a very optimized fashion. Further, Databricks optimized runtime automatizes many aspects and executes user code very efficiently. Additional run-time cost savings can be achieved by using Databricks' optimized connectors, and Databricks AutoML can help reduce model training costs considerably. Databricks also offers auto-scaling which can help prevent over-provisioning.[8]

- Reduction of Server Costs: By using Databrick's serverless architecture, much fine-grained control over resource utilization can be gained. Choosing the optimum managed server pricing can help control analytical solution costs.
- Effort Reduction in Vendor Transition During Termination: A lot of effort is spent by the internal team during vendor termination. Analysis effort can be high if vendors have to be managed, onsite, or near-site presence has to be verified. The use of Knowledge Management modules in Databricks can help utilize the team's existing knowledge and help novice members gain quickly.

Cost optimization or cost-effectiveness is one of the major concerns in big data projects, as the scale of operations is very large. A number of sub-use cases can be defined for this overarching use case, some of which are the following:

9.1. Best Practices for Cost-efficient Data Processing with Databricks

First, become familiar with the relationship between the Databricks runtime version and Apache Spark version, as this dictates the functionalities available and potentially the performance of the Databricks. Next, be mindful of the cluster configuration. The number of worker nodes and their types should correlate with the workload requirements because Databricks bills both for the running of the cluster and the allocated resources within the cluster. Furthermore, though autoscaling may seem like a cost-efficient option, consider that the overhead costs of frequently adding or removing worker nodes may outweigh the benefits. Then, optimize the code. Use appropriate Databricks-specific optimization techniques such as Z-ordering, caching, and optimizing write operations. Finally, monitor the spend and resource usage of the cluster throughout the process because continual assessment enables timely corrections and ensures that you stay within the budget. The Databricks platform is incredibly powerful, but it is only cost-efficient if used mindfully.

The Databricks Unified Data Analytics Platform, from the founders of Apache Spark, unifies data science and engineering to provide an interactive data analytics solution. Because the Databricks platform abstracts the users from the underlying big data and cloud infrastructure complexities, it is easy to overspend where it matters most – dollar ingested, processed, and stored with the data. In this paper, I will examine some best practices for cost-efficient data processing with Databricks.[2]

10. Case Studies of Databricks Implementation in Industries

The proposed solution was implemented in a large retail organization with operations in 30 countries. The study's sample involved nearly 1 million customers with 400 million transactions. The proposed solution allowed for the real-time calculation of CLV and the prediction of the next purchase. The present study identifies the most relevant columns required from a Big Data architecture to predict the next purchase. The proposed solution presents a high-performance level and can easily handle an increasing volume of data. A Big Data architecture that combines Databricks with the GDPR Big Data Reference Architecture was used to manage the significant volume of data and the high velocity of data flow while ensuring the data privacy of customers.

Customer Lifetime Value (CLV) can be used to predict the future value of all customers. Predicting customer behavior and understanding their level of interest is key to providing the best offer and ensuring the most relevant communication. However, there is a tremendous amount of customers and a huge flow of data that needs to be managed. The present study aims to facilitate offer optimization based on a calculation of the CLV, implemented by predicting customer behavior through supervised machine learning models. A Big Data architecture that combines Databricks with the GDPR Big Data Reference Architecture was used to manage the significant volume of data and the high velocity of data flow while ensuring the data privacy of customers.[12]

10.1. Healthcare Industry Case Study

Another great advantage of Databricks in this use case was 'Databricks Delta'. Delta allowed us to write the Apache Spark ETL pipelines in an optimized way. For example, we used the feature called 'Auto Optimize', which automatically coalesces small files for the best-read performance. It also enabled the data team to concentrate on writing the data pipelines without needing to write separate logic on how to manage the historical data in the data lake. The 'Time Travel' feature of Databricks Delta made it easy to address data quality issues and to recover data up to 30 days ago. In the healthcare industry, one of the most important things, besides having fresh data, is data quality. Having the possibility of comparing the data written in the data lake with the data in the source database and easily recover data for specific dates was a crucial advantage of using Databricks in this big data project.

Databricks provides unified data analytics solutions, also known as Apache Spark based solutions, which unite data engineering, data science, and business. I have worked on a POC with a healthcare industry where the main goal was to optimize and automate their current ETL process. This process was mainly executed in DataStage and took around 12 hours for the whole execution flow. With the Apache Spark solution proposed by Databricks, we managed to reduce this time to 3 hours, giving the possibility to the client to have fresh data in their data lake by 10 am. [13]

11. Challenges and Solutions in Implementing Databricks

Aside from that, it is essential to establish stringent governance rules to prevent unmonitored spending when using Databricks in the cloud. A clear governance model that has to be well-communicated with all users involved and enforced by the IT team. Additionally, the secrets management system can be utilized to store information such as a key vault or a Databricks-backed secret scope. This allows easy integration with collaborators' notebooks without exposing sensitive information. To verify collaborator's access to the secret information, a defined set of rules should be established, and periodic auditing has to be done through an access control list (ACL). With these solutions, we can overcome most of the major issues and well implement Databricks across the organization.

Although Databricks greatly simplifies the implementation of big data projects, there are several challenges we may face along the way. First, inefficient or incorrect coding can easily lead to increased costs of running Databricks in the cloud. Moreover, it can be tricky to establish efficient data pipelines for real-time data processing. Additionally, as Databricks is a relatively new technology and there is a shortage of professionals with experience using it. To solve these issues, regular training programs can be implemented to upskill the existing teams. Furthermore, the Databricks platform can be shared with professionals both inside and outside the organization to accumulate knowledge and facilitate quick problem-solving. Lastly, the support team can create and maintain templates of common patterns and best practices to ensure consistency in solution implementation across the board.[10]

11.1. Overcoming Scalability Challenges with Databricks

Databricks allows teams with different skill sets to collaborate using the same underlying tools, removing silos and enabling organizations to build big data pipelines that actually work at big data scale. In this chapter, we present several real-world use cases of Databricks in big data projects. These examples were provided by Databricks customers, and the details have been scrubbed so as to obscure the customer's identity while leaving the core technical content intact. The use cases span a wide array of industries and data types, including ETL for a data warehouse replacement, real-time clickstream analysis, IoT for product quality, social media analytics, and more.

Databricks provides a collaborative environment that sits on top of big data platforms, such as Apache Spark, that are designed for massive scalability and are some of the most widely deployed big data tools. However, these platforms can be challenging to use. Spark has a difficult API, requires a lot of tuning, and can be plagued by long runtimes if not implemented correctly. Furthermore, the environments in which these tools are typically used are often fragmented - data engineers, data scientists, and business analysts often use different tools and have different skill sets, leading to siloed data and sub-optimal big data pipelines.

12. Future Trends and Innovations in Databricks

Databricks started in the big data domain and its core strength till date is to provide a platform which is equally good for data engineering, data science, and machine learning. AI is an obvious next step and Databricks is already well placed there with its ML runtime. Going forward, it is expected to make things more and more easy in the AI space. The roadmap will be to democratize AI and make a few people who are specialized in AI more productive, to enable the masses who are currently not so deeply involved in AI. The specialization will still happen and there will be a few areas where the democratization of AI will not work and the traditional tools with the specialized workforce will continue to be relevant, but the bulk of the organizations will end up using the democratized tools, technologies, and processes to execute AI projects.

Databricks as a big data platform is evolving at a fast pace. It is built on and for the cloud and hence leverages all the ingredients of cloud like scalability, agility, and collaboration. The direction of Databricks is also influenced by the way fast big data is evolving. Features like Delta Lake, MLflow, and Koalas are addressing some of the needs that were felt in the earlier generations. However, fast data is a continuously evolving space and there will always be some gaps to be filled. Some of the broad areas in which Databricks is expected to innovate going forward are more and more automation, democratization of data and data science, easy and more managed machine learning lifecycle, and more and more unification of data and AI.

12.1. AI-driven Enhancements in Databricks

Databricks has an ML runtime – a variant of their usual runtime – specifically designed to make the most common Machine Learning workflows faster and easier without sacrificing generality and compatibility. Quite obviously, all the popular AI libraries are pre-installed in that runtime. After all, Databricks was born out of the design and implementation of Apache Spark, the Big Data technology that has seen the most – and most fruitful, and most tumultuous – interactions with the Machine Learning and Deep Learning codes and communities, over the years. The tight integration between Spark and the most common AI frameworks (Spark has an MLlib library that is quite performant for a limited set of algorithms) is one of the key selling points of Databricks, and is explicitly cited in any introductory materials.

One key unmistakable trend is the massive adoption of AI techniques – machine learning in particular – in any sort of Big Data project. Big Data technologies are used to handle a huge mass of data – clearly the precondition for proper training of advanced Artificial Intelligence models, and then AI models are the perfect tool to extract ultimate value from the data you're storing and managing with Big Data technologies. Rinse and repeat. It's no surprise, then, that every Big Data technology you're considering will boast several out-of-the-box integrations with popular AI libraries – whether that's a good idea or not, is a topic for another day – and pipelines connecting Big Data to AI frameworks.[6]

13. Conclusion

In summary, Databricks has established itself as a vital tool in the landscape of big data analytics, providing a unified platform that enhances collaboration, scalability, and performance across various industries. Through its seamless integration with cloud services and robust support for Apache Spark, Databricks simplifies the complexities of big data processing and machine learning workflows. The platform's features such as Delta Lake, MLflow, and Koalas address critical needs in data management, while its serverless architecture and optimized runtime ensure efficient resource utilization and cost savings.

Real-world applications of Databricks demonstrate its versatility and effectiveness in addressing diverse business challenges, from real-time data processing and predictive analytics to cost optimization and data governance. The case studies highlighted in this paper underscore Databricks' ability to transform data into actionable insights, driving innovation and competitive advantage for organizations worldwide.

Looking forward, Databricks is poised to continue its evolution with a strong focus on democratizing AI, enhancing automation, and unifying data science and machine learning processes. As the big data ecosystem evolves, Databricks is well-positioned to meet the demands of fast-paced data environments, ensuring that organizations can leverage their data assets to the fullest.

By bridging the gap between data engineering and data science, Databricks not only simplifies the technical complexities of big data projects but also fosters a collaborative and productive environment for teams. This holistic approach to data analytics and machine learning sets the stage for continued advancements and success in the realm of big data.[3]

References

[1]. E. Niemi and S. Pekkola, "The benefits of enterprise architecture in organizational transformation," *Business & Information Systems Engineering*, vol. 62, no. 6, pp. 517-527, 2020. [Online]. Available: https://link.springer.com/article/10.1007/s12599-020-00646-3

[2]. Databricks, "New Study: Databricks Delivers Nearly $29 Million in Economic Benefits and Pays for Itself in Less Than Six Months," *Databricks Blog*, Apr. 28, 2020. [Online]. Available: https://databricks.com/blog/2020/04/28/new-study-databricks-delivers-nearly-29-million-in-economic-benefits-and-pays-for-itself-in-less-than-six-months.html

[3]. F. Saleem and B. Fakieh, "Enterprise architecture and organizational benefits: a case study," *Sustainability*, vol. 12, no. 15, p. 5943, 2020. [Online]. Available: https://www.mdpi.com/2071-1050/12/15/5943

[4]. D. Dumitriu and M. A. M. Popescu, "Enterprise architecture framework design in IT management," *Procedia Manufacturing*, vol. 46, pp. 134-141, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2351978920301344

[5]. Y. Gong and M. Janssen, "Roles and capabilities of enterprise architecture in big data analytics technology adoption and implementation," *Journal of Theoretical and Applied Electronic Commerce Research*, vol. 16, no. 3, pp. 302-315, 2021. [Online]. Available: https://www.mdpi.com/0718-1876/16/3/34

[6]. M. P. Uysal and A. E. Mergen, "Smart manufacturing in intelligent digital mesh: Integration of enterprise architecture and software product line engineering," *Journal of Industrial Information Integration*, vol. 23, p. 100223, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2452414X2100041X

[7]. M. M. Alvord, F. Lu, B. Du, and C. A. Chen, "Big data fabric architecture: How big data and data management frameworks converge to bring a new generation of competitive advantage for enterprises," *Enterprise Architecture Professional Journal*, vol. 5, no. 2, 2022. [Online]. Available: https://eapj.org/big-data-fabric-architecture/

[8]. P. Ghavami, "Big data management: Data governance principles for big data analytics," *Big Data Management*, 2020. [Online]. Available: https://bigdatamanagement.com/

[9]. M. Armbrust, T. Das, L. Sun, B. Yavuz, S. Zhu, et al., "Delta lake: high-performance ACID table storage over cloud object stores," in *Proceedings of the 2020 SIGMOD*, 2020, pp. 2395-2409. [Online]. Available: https://dl.acm.org/doi/10.1145/3318464.3386134

[10]. R. K. Batwada, N. Mittal, and E. S. Pilli, "Uncovering Data Warehouse Issues and Challenges in Big Data Management," in *Big Data Machine Learning, and Analytics: A Perspective on the Industry and Academia*, Springer, 2020, pp. 133-152. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-57083-1_9

[11]. D. Oreščanin and T. Hlupić, "Data lakehouse-a novel step in analytics architecture," in *2021 44th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, 2021, pp. 1331-1335. [Online]. Available: https://ieeexplore.ieee.org/document/9456653

[12]. M. Armbrust, A. Ghodsi, R. Xin, et al., "Lakehouse: a new generation of open platforms that unify data warehousing and advanced analytics," *Proceedings of the VLDB Endowment*, vol. 14, no. 12, pp. 2481-2494, 2021. [Online]. Available: https://vldb.org/pvldb/vol14/p2481-armbrust.pdf

[13]. S. Cisneros-Cabrera, A. V. Michailidou, S. Sampaio, et al., "Experimenting with big data computing for scaling data quality-aware query processing," *Expert Systems with Applications*, vol. 177, p. 114906, 2021. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S095741742100391X

[14]. K. H. Hu, M. F. Hsu, F. H. Chen, et al., "Identifying the key factors of subsidiary supervision and management using an innovative hybrid architecture in a big data environment," *Financial Innovation*, vol. 7, no. 1, p. 42, 2021. [Online]. Available: https://jfin-swufe.springeropen.com/articles/10.1186/s40854-021-00242-4