Fundamentals of Accelerated Data Science

NVIDIA

Workshop Overview

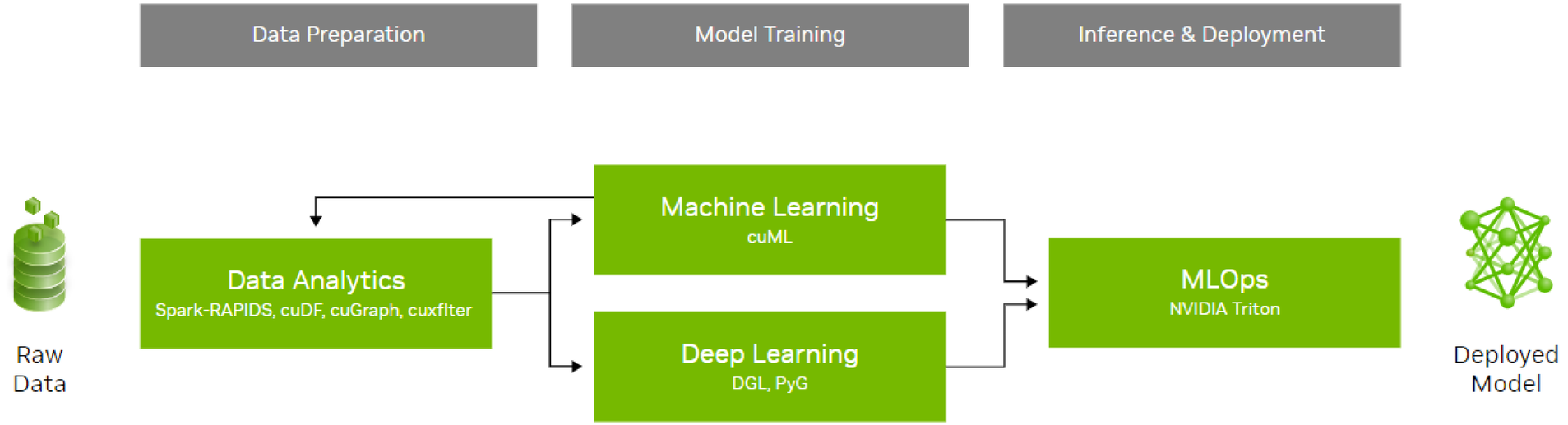
An introduction to data science with a focus on speed and efficiency

Learning objectives:

- Understand the fundamental concepts of data science and parallel computing
- Explore practical examples of accelerated data science pipelines
- Examine the methods used to achieve acceleration in data science and discuss their broader implications

This workshop will not cover statistical analysis, neural networks, and distributed computing

	Workshop Outline
Task 1	Data science overview Data manipulation
Task 2	Graph analytics
Task 3	Machine Learning
Coding Assessment	Biodefense



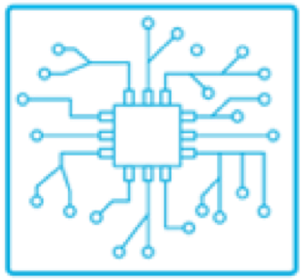
Section 3 Agenda

- Machine Learning Fundamentals
- End-To-End Machine Learning
- Accelerating Machine Learning
- Hands-On Lab

Machine Learning

Data, algorithm, and compute

- Data science has a broader focus that encompasses various techniques for extracting insights and meaning from data, including statistical analysis and data visualization. In contrast, machine learning primarily focuses on building algorithms that enable computers to learn from data and make predictions.
- Pillars of machine learning:



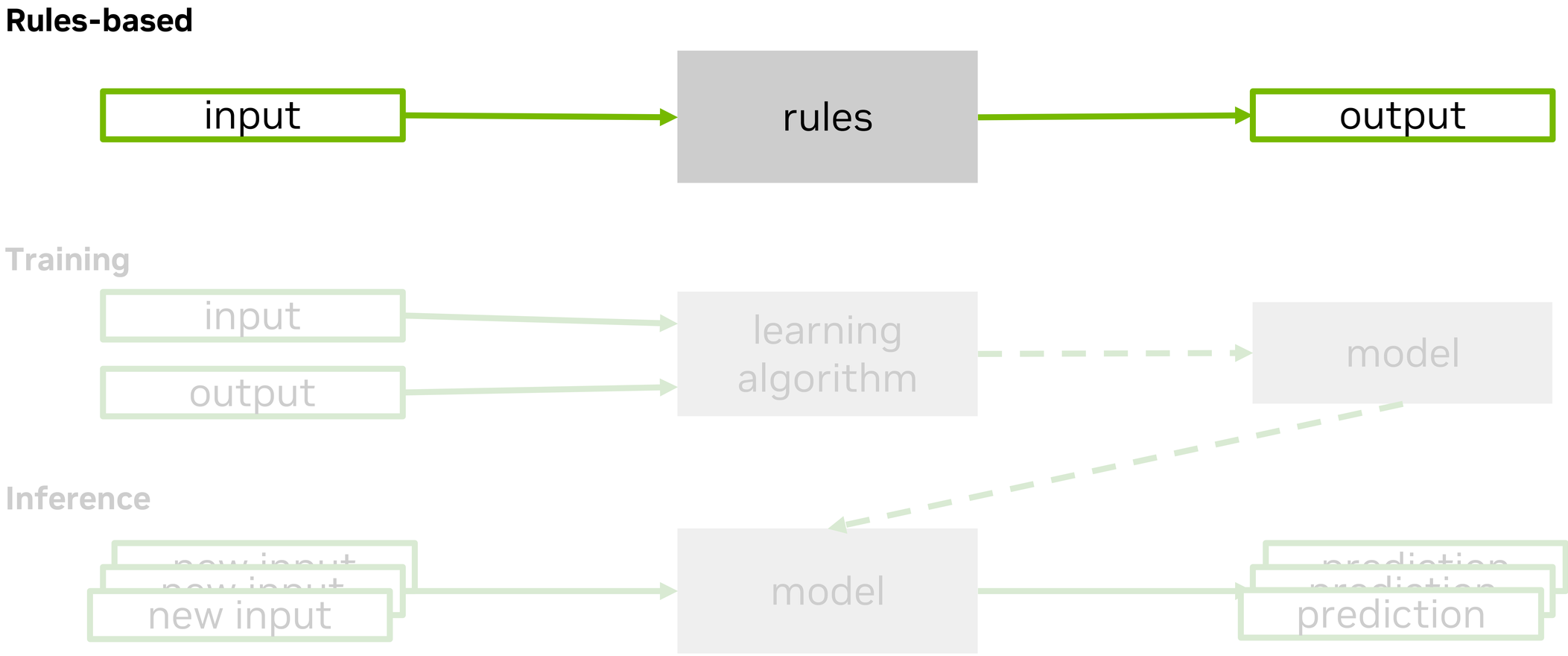
Computing Power

Algorithm Power

Data Availability

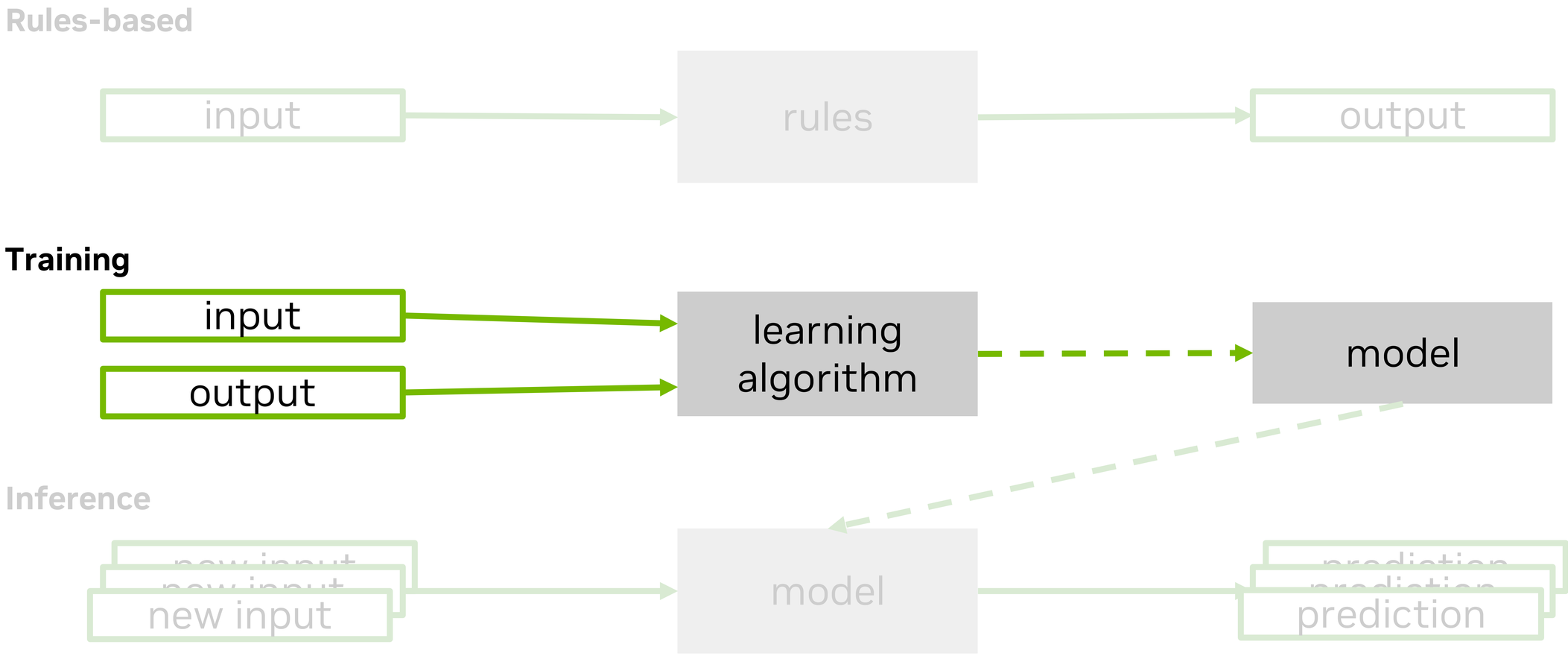
Traditional Programming: Rules-Based

Explicitly define rules and logic to follow



Model Training

Models learn patterns from data to make predictions or decisions without explicit programming

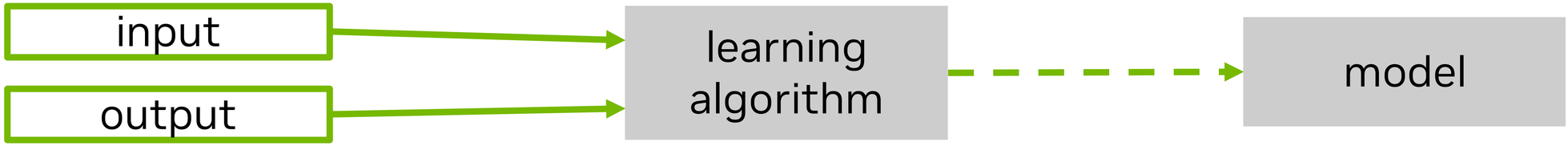


Supervised vs. Unsupervised

Trained on labeled data to make predictions or classifications on new, unseen data

Rules-based

Training



Supervised

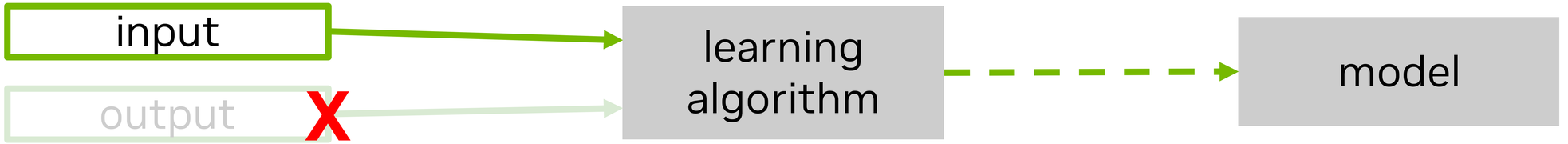
- **Regression**: predicting a continuous value
- **Classification**: predicting a discrete label

Supervised vs. **Unsupervised**

Identifying unknown patterns and structures in data without predefined categories

Rules-based

Training



Supervised

- **Regression**: predicting a continuous value
- **Classification**: predicting a discrete label

Unsupervised

- **Clustering**: grouping data points together
- **Dimensionality reduction** while retaining information

Example Dataset

Multiple machine learning tasks can be applicable to the same dataset

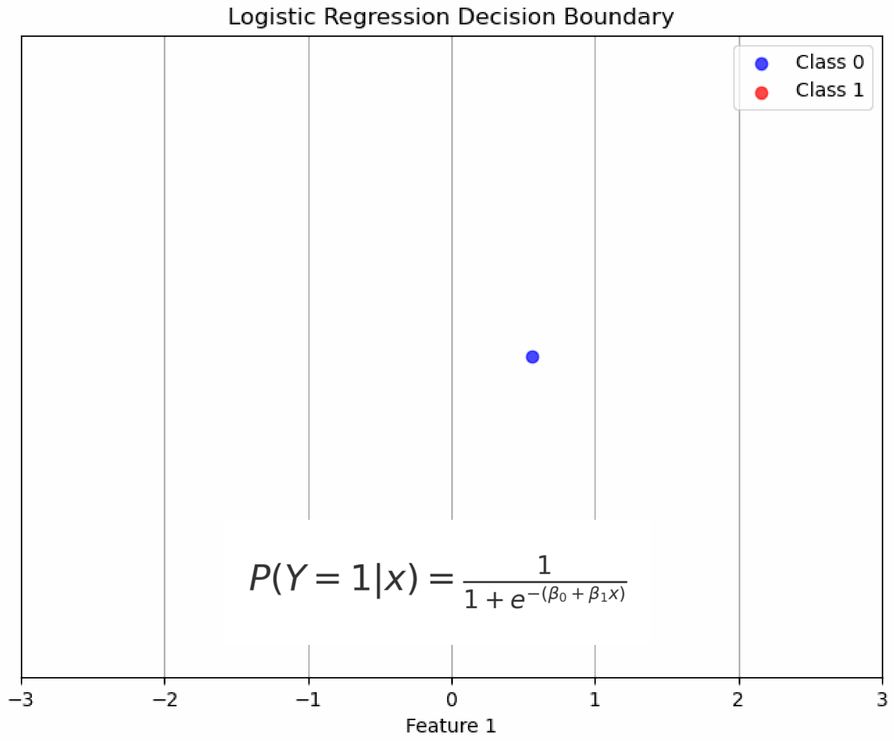
- Features
- Target

	Feature 1	Feature 2	Class
0	0.559	2.389	0
1	1.312	-0.717	1
2	-1.56	-1.925	0
3	-2.281	-0.137	0
4	1.561	-0.428	1
5	-0.808	1.197	0
6	-0.271	-2.256	1
7	0.481	0.549	1
8	-1.208	-1.269	0
9	...		

Example Dataset

Classification assigns data to known categories

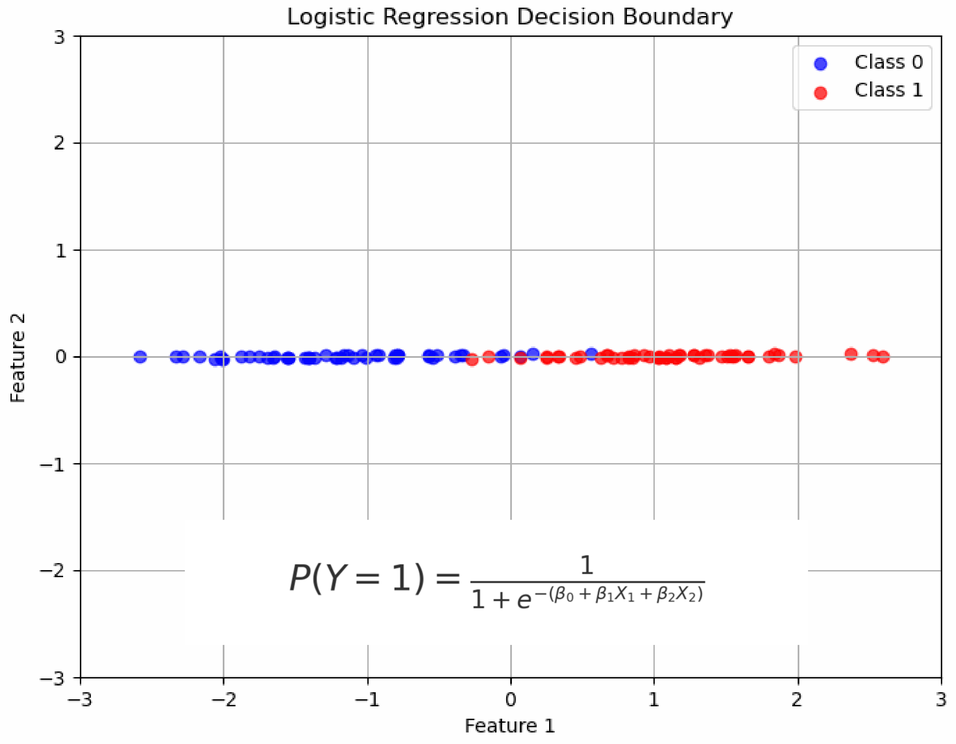
	Feature 1		Class
0	0.559		0
1	1.312		1



Example Dataset

Classification assigns data to known categories

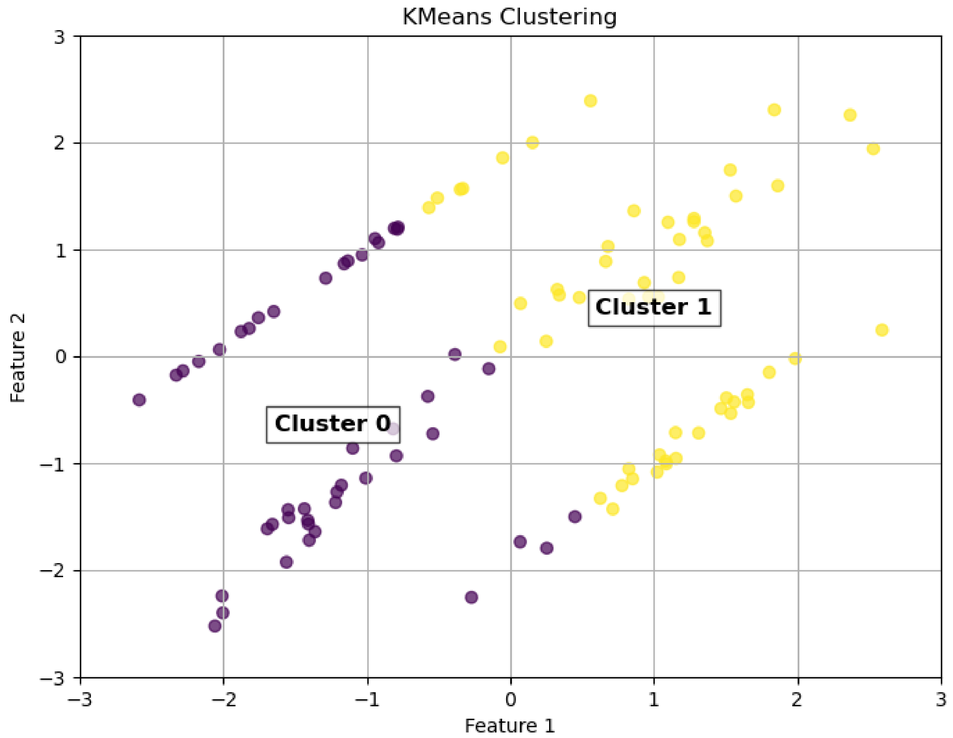
	Feature 1	Feature 2	Class
0	0.559	2.389	0
1	1.312	-0.717	1
2	-1.56	-1.925	0
3	-2.281	-0.137	0
4	1.561	-0.428	1
5	-0.808	1.197	0
6	-0.271	-2.256	1
7	0.481	0.549	1
8	-1.208	-1.269	0
9	...		



Example Dataset

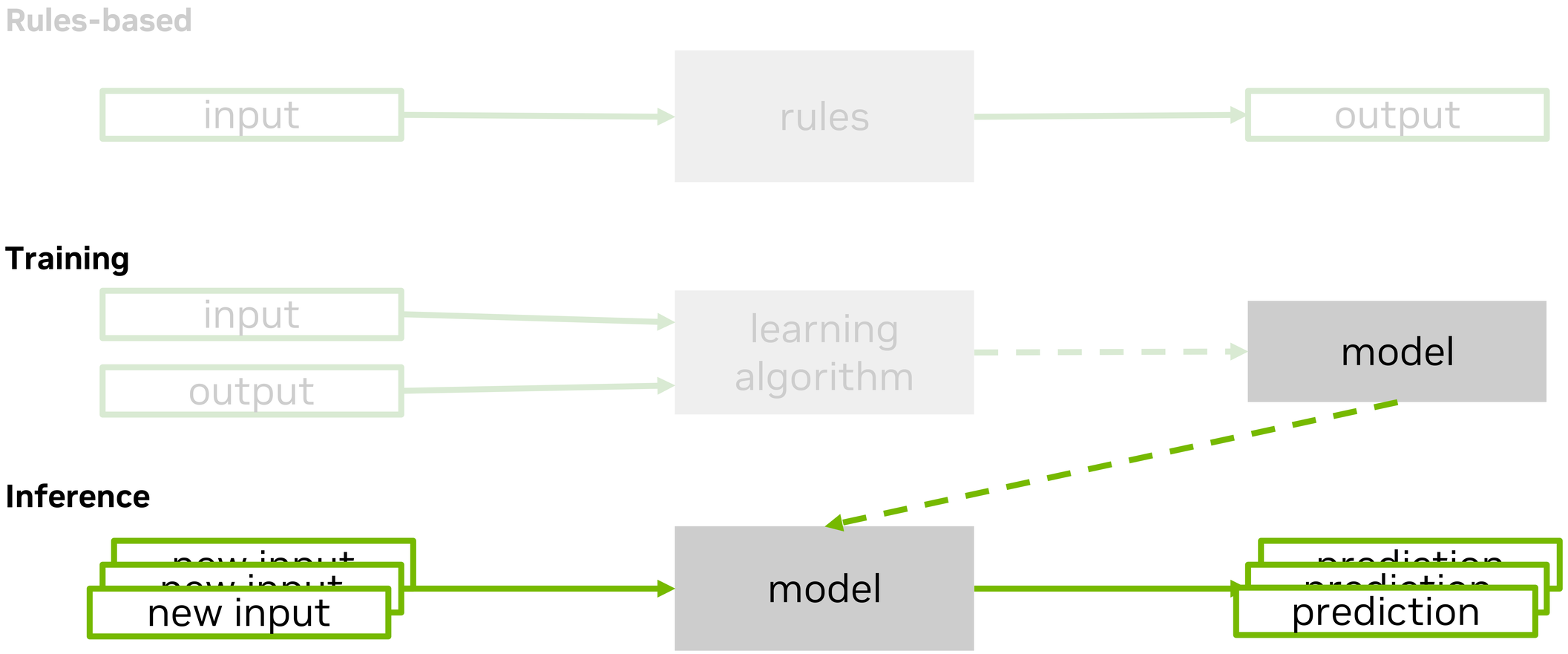
Clustering discovers inherent groups in data without prior labeling

	Feature 1	Feature 2	
0	0.559	2.389	
1	1.312	-0.717	
2	-1.56	-1.925	
3	-2.281	-0.137	
4	1.561	-0.428	
5	-0.808	1.197	
6	-0.271	-2.256	
7	0.481	0.549	
8	-1.208	-1.269	
9	...		



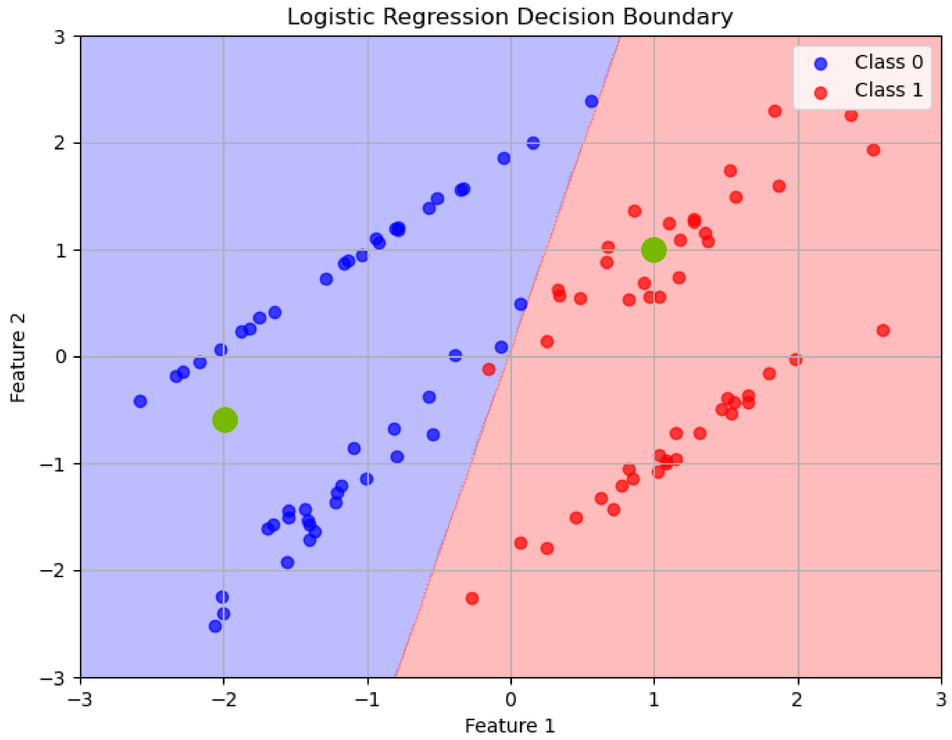
Inference

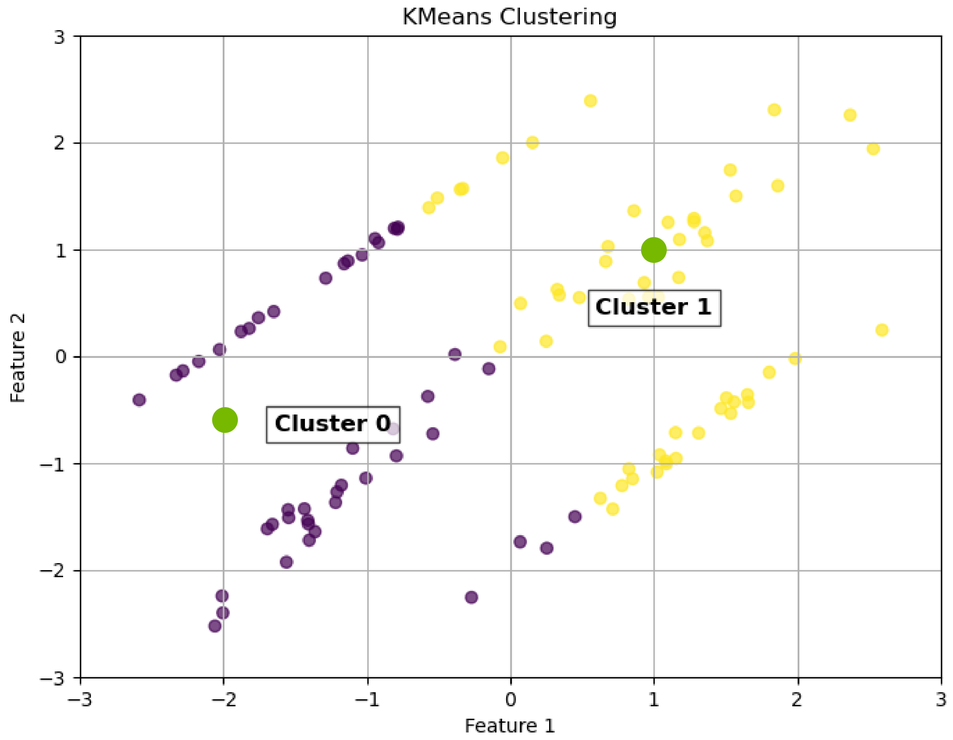
Process incoming data and producing results based on patterns learned during training



Example Dataset

Process incoming data and producing results based on patterns learned during training



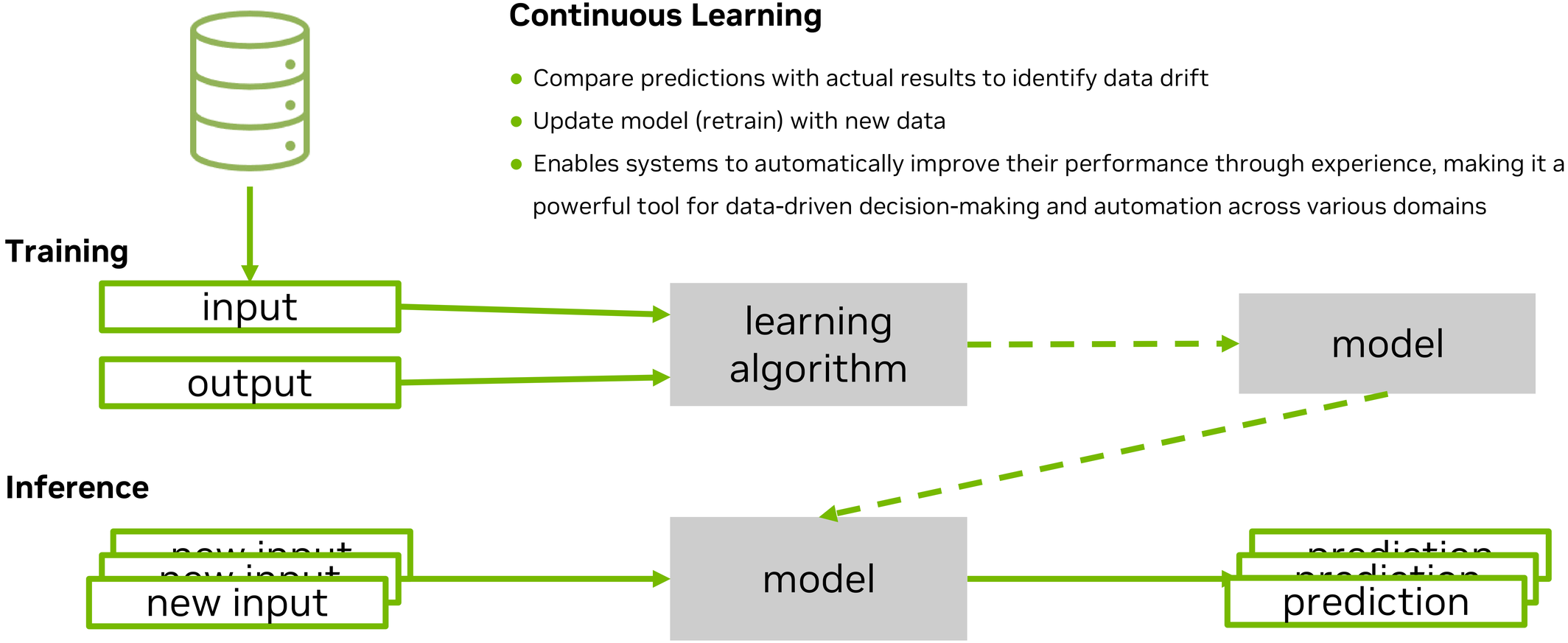


Lifelong Learning

Allows models to update and improve their performance as new data becomes available

Continuous Learning

- Compare predictions with actual results to identify data drift
- Update model (retrain) with new data
- Enables systems to automatically improve their performance through experience, making it a powerful tool for data-driven decision-making and automation across various domains



Common Use Cases

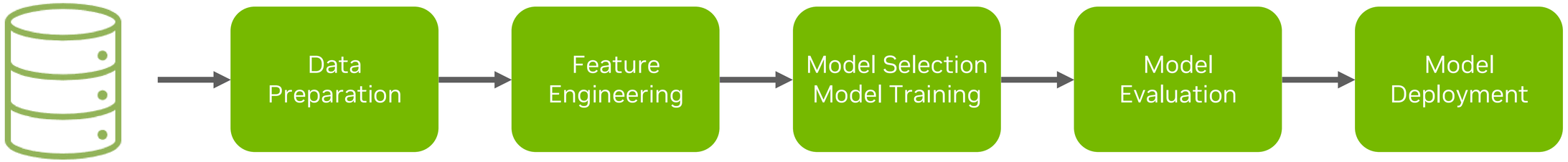
	Supervised	Unsupervised
Online Services	• Spam / Not Spam • Predict Probability Customer Will Click on Ad	• Group Articles into Categories • Similar Search Results
Financial Services	• Credit Card Fraud	• Anomaly Detection
Telecom	• Predict Customer Churn	
Health Care	• Probability of Readmission • Predict Days of Hospital Stay	• Patient Similarity
Real Estate	• Predict House Price	
Retail	• Prejudice Price • Forecast Sales • Sentiment Analysis	• Similar Customers • Product Similarity • Customer Group • Products Which Are Purchased Together

Benefits:

- **Problem solving**: excels at complex problems where patterns are not easily discernible
- **Adaptability**: models can adapt to new data, new tasks, and improve over time with retraining

Machine Learning Model Development Workflow

Systematic process of developing, training, evaluating, and deploying machine learning models



- Define the business problem and project objectives
- Gather data from relevant sources

- Exploratory data analysis (ET)
- Clean, transform, and prepare data for analysis

- Format data for model consumption

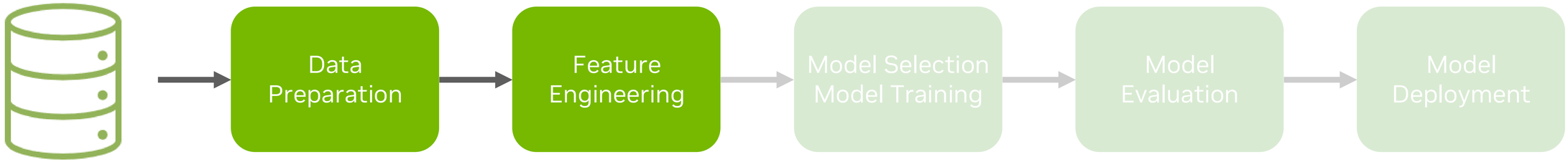
- Choose appropriate algorithms based on the problem and data

- Assess the model's performance using various metrics

- Integrate the model into production systems
- Continuously monitor the model's performance and retrain as needed

Machine Learning Model Development Workflow

Systematic process of developing, training, evaluating, and deploying machine learning models



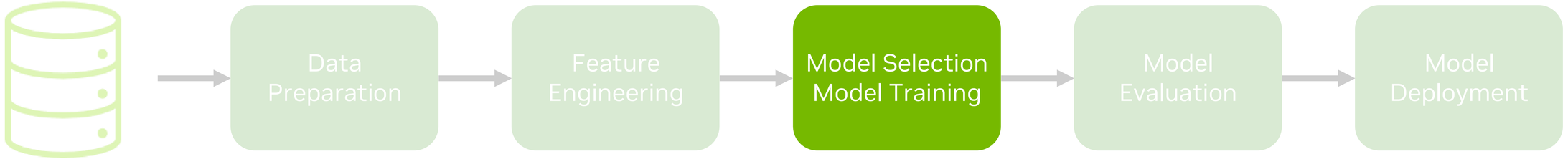
- **Data preparation** – clean, transform, and prepare the data
- **Feature engineering** is often considered one of the most crucial and impactful steps in the machine learning process to improve a model's predictive power and ability to generalize to new data
  - Transforms raw data into a format that better represents the underlying problem, making it easier for algorithms to learn patterns and make predictions
  - Feature engineering is an iterative process that allows for continuous refinement and improvement of model inputs

Use cuDF to create features for downstream ML tasks:

- cuML
- XGBoost
- cuDNN

Machine Learning Model Development Workflow

Systematic process of developing, training, evaluating, and deploying machine learning models



- **Model selection** – choosing the appropriate algorithm and configuration(s)
- **Model training** – fitting the model to the dataset
  - Faster runtime shortens development time and improves productivity

Task	Goal	Algorithm / Model
Classification	Assign input data to predefined categories or labels	Logistic Regression, Decision Trees, Random Forest, SVM, KNN, Naïve Bayes, Neural Networks
Regression	Predict continuous numerical values based on input features	Linear Regression, Ridge & Lasso Regression, Decision Trees, Random Forest, SVR, Neural Networks, XGBoost
Clustering	Group similar data points together without predefined labels	k-Means, Hierarchical Clustering, DBSCAN, Gaussian Mixture Models (GMM), Mean Shift
Decomposition & Dimensionality Reduction	Reduce the number of input features while preserving important information	PCA, SVD, t-SNE, UMAP, Autoencoders

Classification

Categorize input data into predefined classes

	K-Nearest Neighbors	Logistic Regression	Decision Tree
Approach	KNN makes predictions based on the similarity of new data points to known examples in the training set. The algorithm calculates the distance between the new point and all points in the training set. It identifies the K nearest neighbors and uses majority voting among the K neighbors	Logistic regression is a statistical method used for binary classification. It predicts the probability of an instance belonging to a particular class, using the logistic function (sigmoid) to map predictions to probabilities between 0 and 1	Creates a tree-like model of decisions based on features in the data. From the root node, it splits the data into subsets based on the most significant attributes. The splitting criteria is determined by maximizing information gain or minimizing impurity
Pros	• Makes no assumption about data distribution • “Lazy” learning algorithm • Adapts quickly to new data	• Generally efficient • Less prone to overfitting	• Does not require feature scaling • Robust to outliers • Can handle categorical variables • Makes no assumption about data distribution
Cons	• Prone to overfitting • Sensitive to outliers • Requires feature scaling • High computational cost for large datasets, especially during prediction • High memory requirement (needs to store the entire training dataset)	• Sensitive to outliers • Assumes linear relationship between features and log-odds of outcome • Suitable for binary outcome • Assumes no correlation between features • Requires irrelevant features to be excluded • Can overfit on high-dimensional datasets	• Prone to overfitting • Computationally expensive for large datasets

Decision Tree

Decision trees are prone to overfitting

Recursively splitting the data based on features to create a tree-like structure

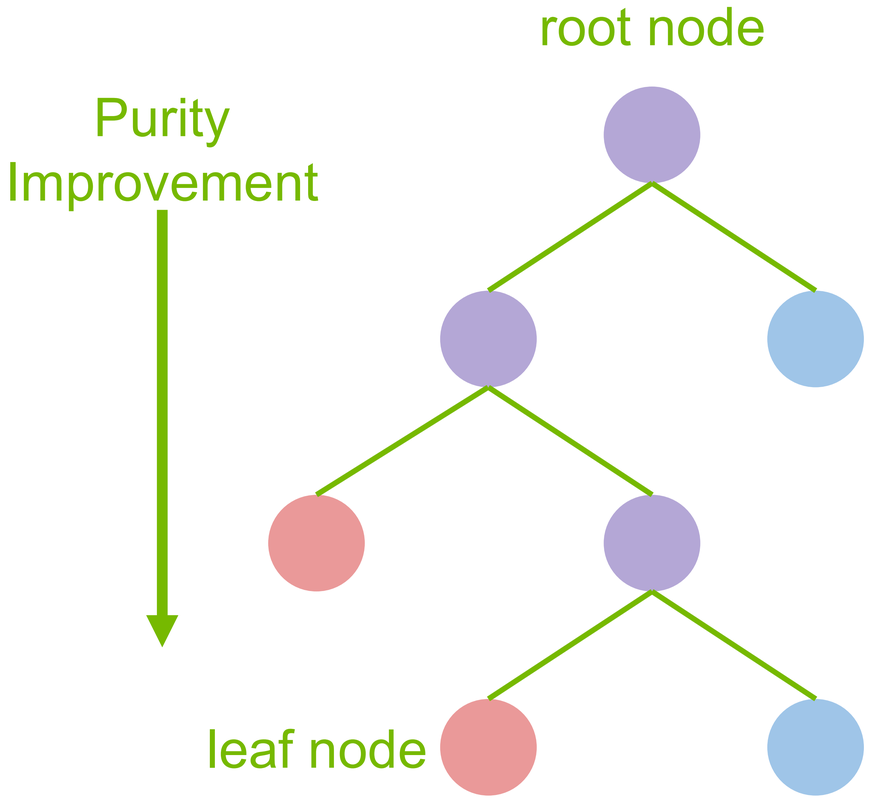
- Select the best feature to split the data to increase the purity of the resulting subsets with respect to the target variable

Challenges:

- Prone to overfitting

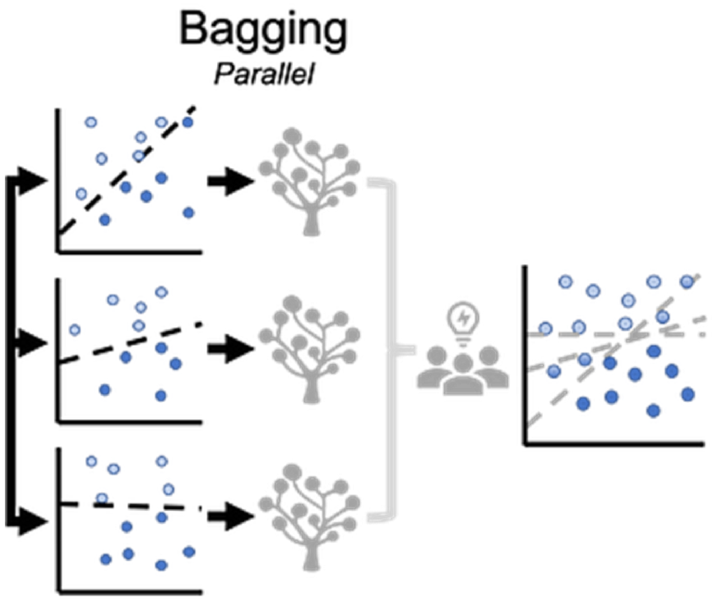
Solution:

- Pruning
- Ensemble



Ensemble

Using multiple decision trees



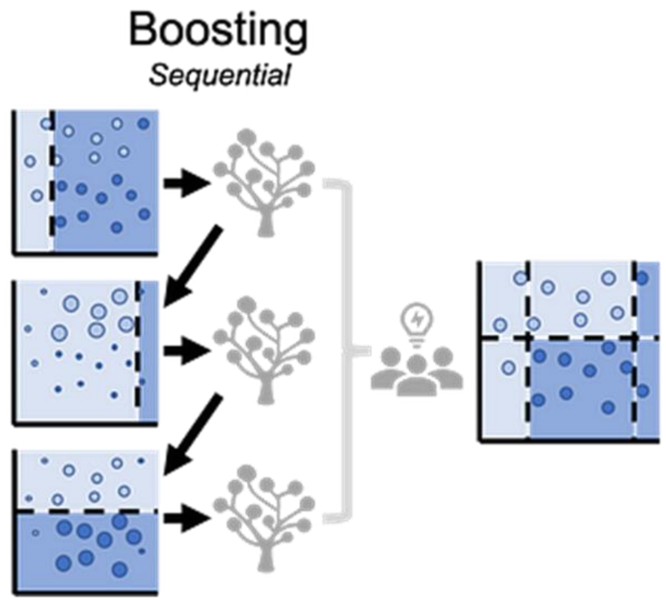
(Random Forest) Bagging is an ensemble technique used to reduce the variance of predictions by considering results from multiple decision tree models that were trained on different sub-samples of the same data set

(Gradient Boosting Decision Tree) Boosting is another type of ensemble learning that combines weak learners to achieve improved model performance. Weak learners are simple models that predict relatively poorly. The concept of boosting is to sequentially train models, each time trying to fit better than before using residual errors

Ensemble

Using multiple decision trees

(Random Forest) Bagging is an ensemble technique used to reduce the variance of predictions by considering results from multiple decision tree models that were trained on different sub-samples of the same data set



(Gradient Boosting Decision Tree) Boosting is another type of ensemble learning that combines weak learners to achieve improved model performance. Weak learners are simple models that predict relatively poorly. The concept of boosting is to sequentially train models, each time trying to fit better than before using residual errors

Clustering

Group unlabeled examples based on their similarity to each other

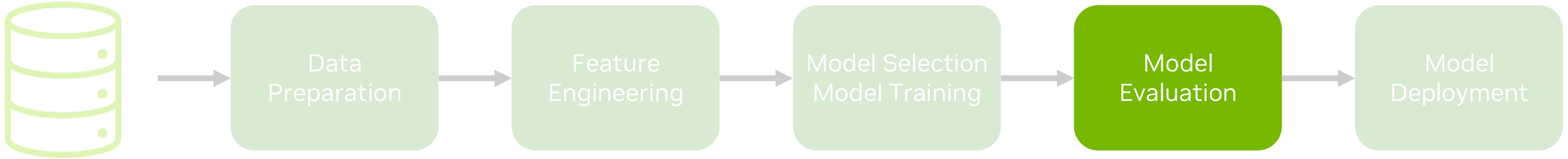
Use cases:

- Market segmentation
- Anomaly detection
- *Data Imputation*

	K-Means	DBSCAN
Approach	Partitions data by iteratively moving cluster centers and re-assigning data points based on which center is closest to the point	Uses spatial density to cluster—finds cluster points within at most epsilon distance of another point in the cluster
Pros	• Fast, simple, and easy to understand • Scalable to large datasets	• Can identify clusters of any arbitrary shapes • Automatically determines the number of clusters • Robust to outliers
Cons	• Assumes spherical clusters • Requires specifying the number of clusters in advance • Sensitive to outliers	• Less efficient for very large datasets • Requires specifying minimum points and maximum neighborhood distance

Machine Learning Model Development Workflow

Systematic process of developing, training, evaluating, and deploying machine learning models



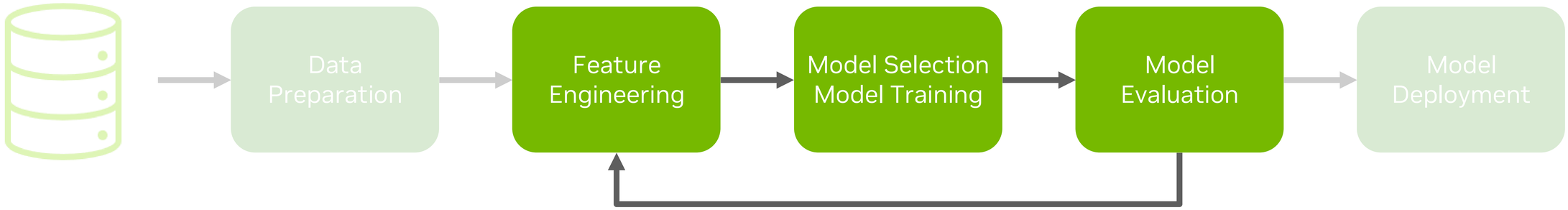
- **Model evaluation** – assessing the quality and performance of a trained machine learning model using various metrics
  - Balance accuracy and inference speed
  - Maximize throughput given an application's accuracy requirement and latency budget

Evaluation metrics:

- **Accuracy**
  - Underfit vs overfit
  - Cross validation is a resampling method to protect against overfitting
- **Latency** and **throughput**
  - Evaluate how well the inference performance scales with increasing data volumes or concurrent requests
  - Ensure that optimizations for speed don't significantly degrade the model's accuracy or performance on its intended task

Machine Learning Model Development Workflow

Systematic process of developing, training, evaluating, and deploying machine learning models



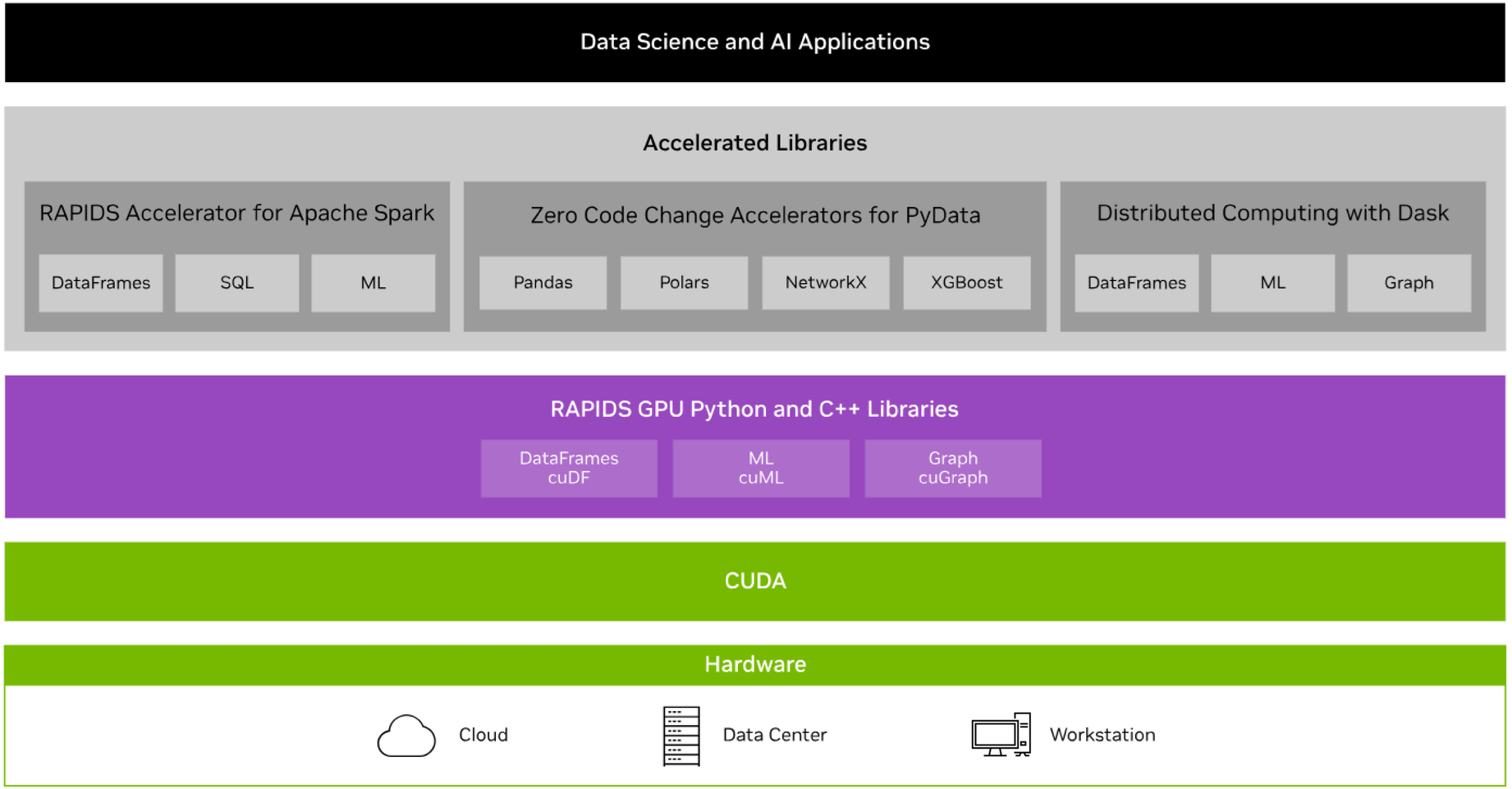
- **Model tuning**
  - Feature selection finds the optimal subset of features
  - Hyperparameter tuning using techniques such as **grid search**

Use cuML to accelerate machine learning model training

- Low data processing cycle time translates to higher productivity, more experimentation, and improved model performance

RAPIDS Transforms Data Science

An optimized hardware-to-software stack for the entire data science pipeline



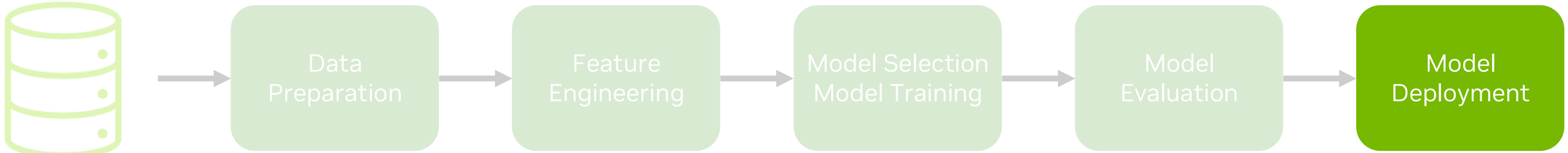
XGBoost + RAPIDS

Optimized gradient boosting machine learning algorithm

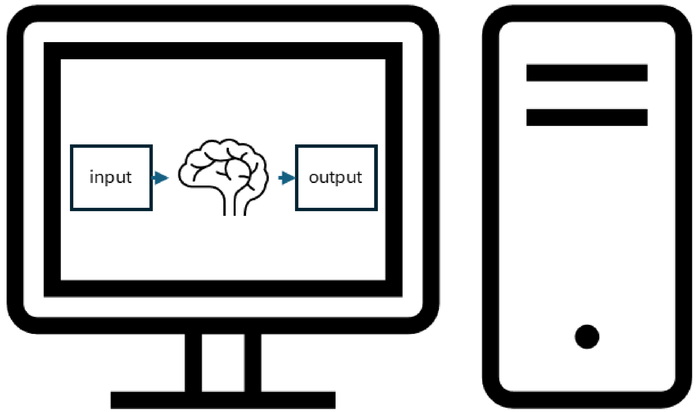
- Extreme Gradient Boosting, is a scalable, distributed gradient-boosted decision tree machine learning algorithm
- Gold standard in single model performance for classification and regression
  - Delivers excellent accuracy and efficiency
  - Handles large datasets effectively through multi-threading
  - Handles imbalanced datasets and outliers
  - Uses regularization to prevent overfitting and improve model generalization
- RAPIDS was built from the start to integrate with this amazing algorithm, including seamless GPU-acceleration such as loading data from cuDF DataFrames and cuPy arrays - this significantly speeds up model training and improves accuracy for better predictions

Machine Learning Model Development Workflow

Systematic process of developing, training, evaluating, and deploying machine learning models



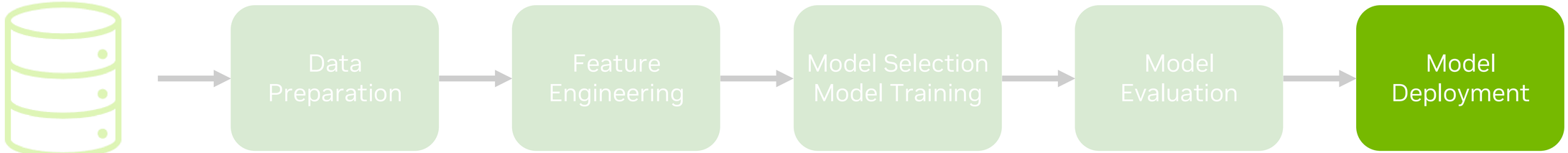
On-Device Inference



Server-Based Inference

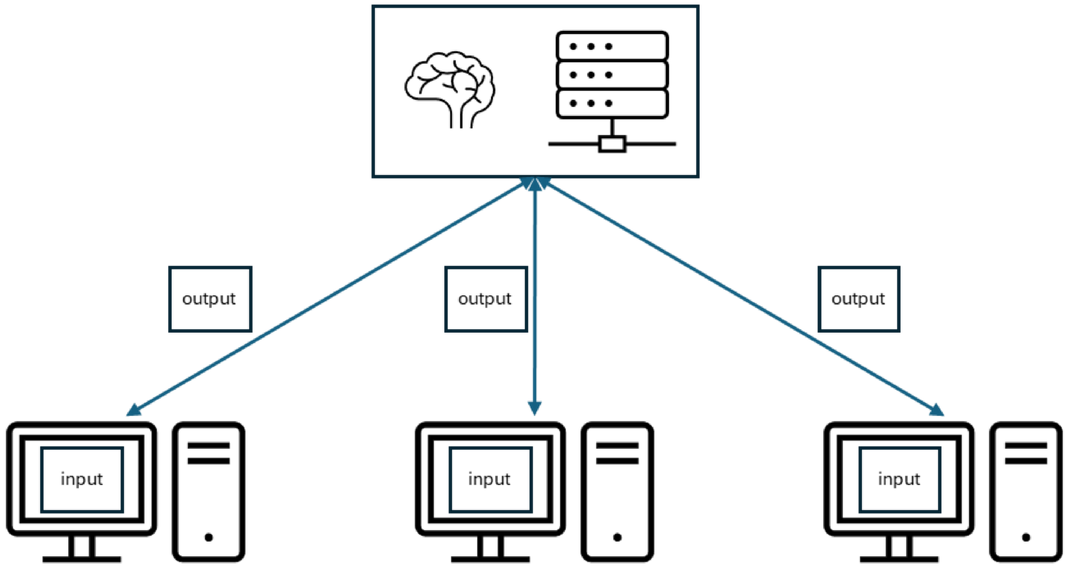
Machine Learning Model Development Workflow

Systematic process of developing, training, evaluating, and deploying machine learning models



On-Device Inference

Server-Based Inference



Triton Inference Server

Open-source inference serving software that is designed for deploying AI models at scale

Key Features:

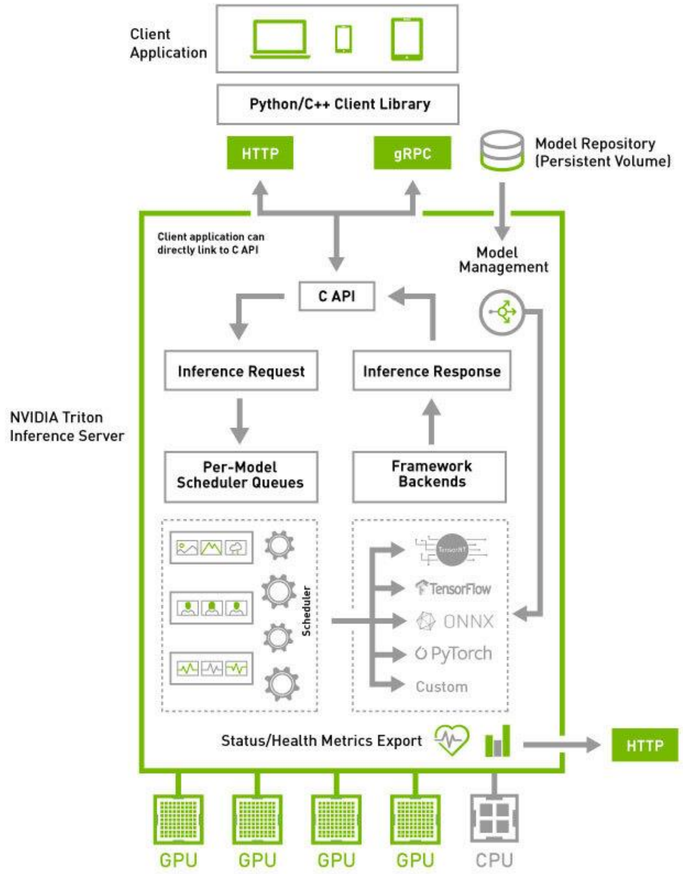
- Multi-framework support (TensorFlow, PyTorch, ONNX, etc.)
- Concurrent model execution
- Dynamic batching for improved performance
- Model versioning and ensembling

Benefits for Data Scientists:

- Simplifies model deployment in production environments
- Enables efficient scaling of machine learning models
- Provides a consistent interface across different ML frameworks

Optimizing Performance:

- Performance analyzer
- Model analyzer



Example of Model Analyzer Report

Online Result Summary

Model: resnet50_libtorch

GPU(s): 1 x NVIDIA TITAN RTX

Total Available GPU Memory: 23.7 GB

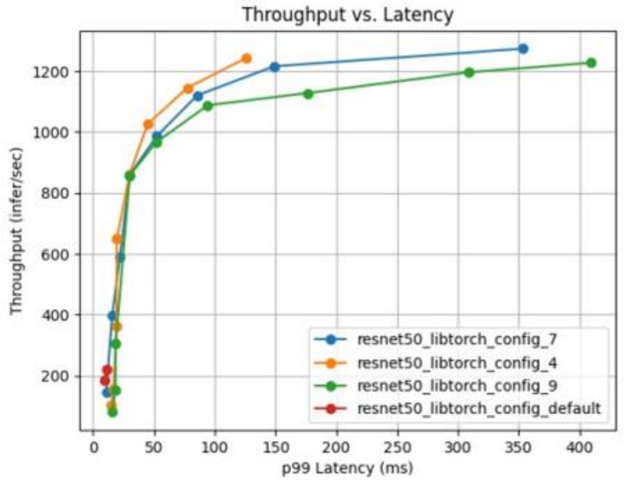
Constraint targets: None

In 46 measurements across 12 configurations, **resnet50_libtorch_config_7** provides the best throughput: **1273 infer/sec**.

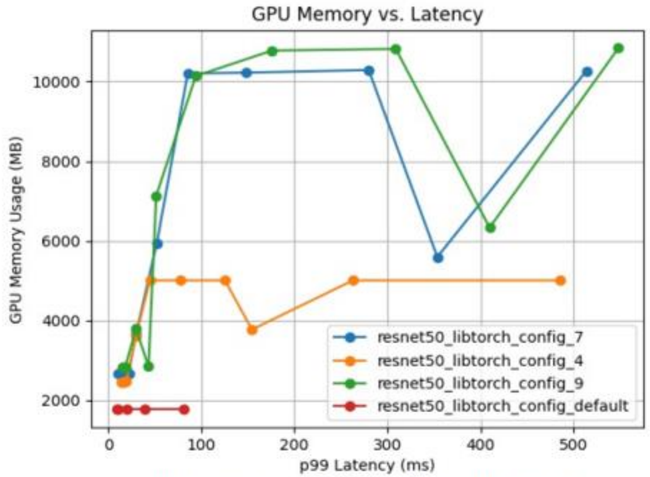
This is a **473% gain** over the default configuration (222 infer/sec), under the given constraints on GPU(s) 1 x NVIDIA TITAN RTX.

- **resnet50_libtorch_config_7**: 6 GPU instances with a max batch size of 32 on platform pytorch_libtorch

Curves corresponding to the 3 best model configuration(s) out of a total of 12 are shown in the plots.



Throughput vs. Latency curves for 3 best configurations.



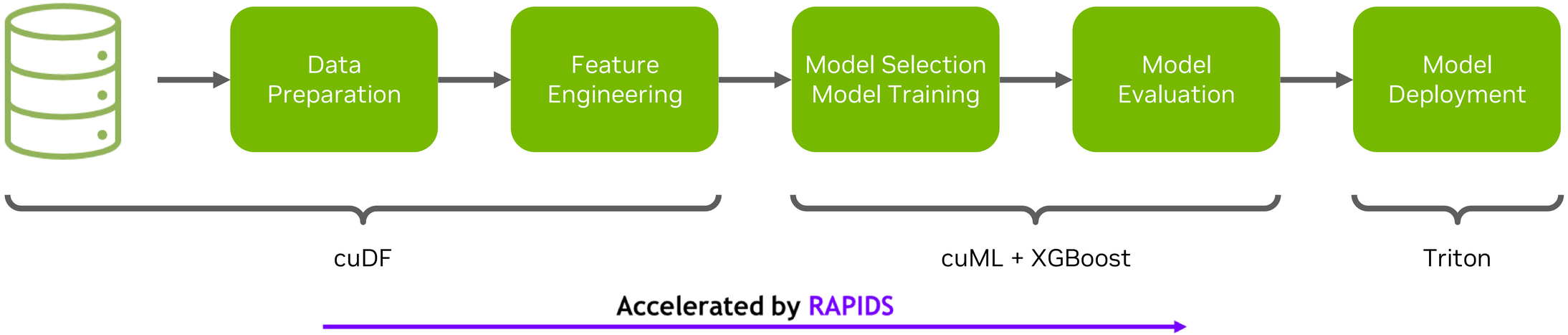
GPU Memory vs. Latency curves for 3 best configurations.

The following table summarizes each configuration at the measurement that optimizes the desired metrics under the given constraints.

Model Config Name	Max Batch Size	Dynamic Batching	Total Instance Count	p99 Latency (ms)	Throughput (infer/sec)	Max GPU Memory Usage (MB)	Average GPU Utilization (%)
resnet50_libtorch_config_7	32	Enabled	6:GPU	353.539	1273.38	5598	94.3
resnet50_libtorch_config_4	16	Enabled	5:GPU	125.635	1242.25	5011	90.2
resnet50_libtorch_config_9	32	Enabled	7:GPU	409.616	1227.02	6334	95.3
resnet50_libtorch_config_default	128	Disabled	1:GPU	10.938	221.846	1773	41.0

Machine Learning Model Development Workflow

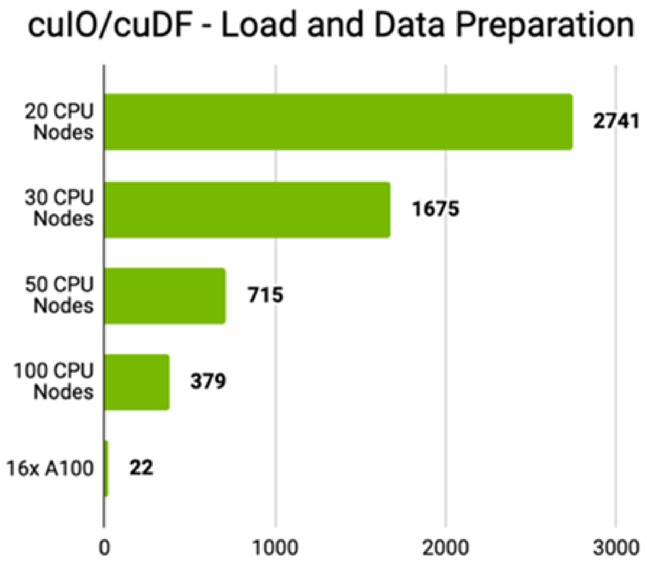
Systematic process of developing, training, evaluating, and deploying machine learning models

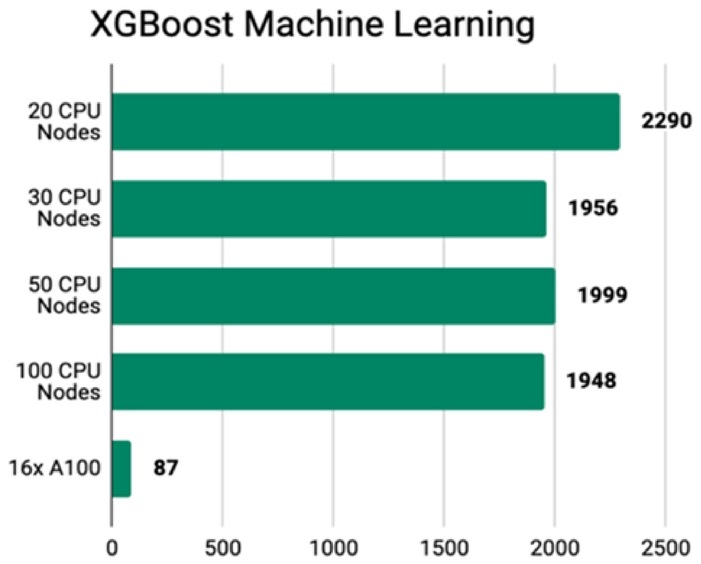


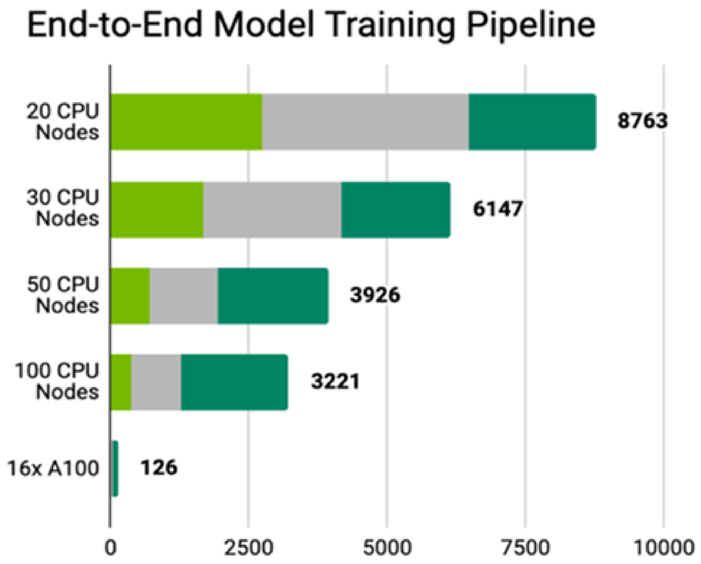
- Efficient data processing on large datasets leads to better results
  - Facilitates higher data quality and quantity that directly impact model performance
  - Increases productivity
  - Enables more experimentation for model tuning
  - Fosters innovation and progress
  - Reduces time to insight and total cost of ownership
- Seamlessly integrate with existing workflows through plugins
- Optimal model deployment that balances performance and resource allocation

Proven Faster for Data Science

Realize efficiency and scalability with multi-GPU for massive datasets







Benchmark

200GB CSV dataset; Data preparation includes joins, variable transformations.

CPU Cluster Configuration

CPU nodes (61 GiB of memory, 8 vCPUs, 64-bit platform), Apache Spark

DGX Cluster Configuration

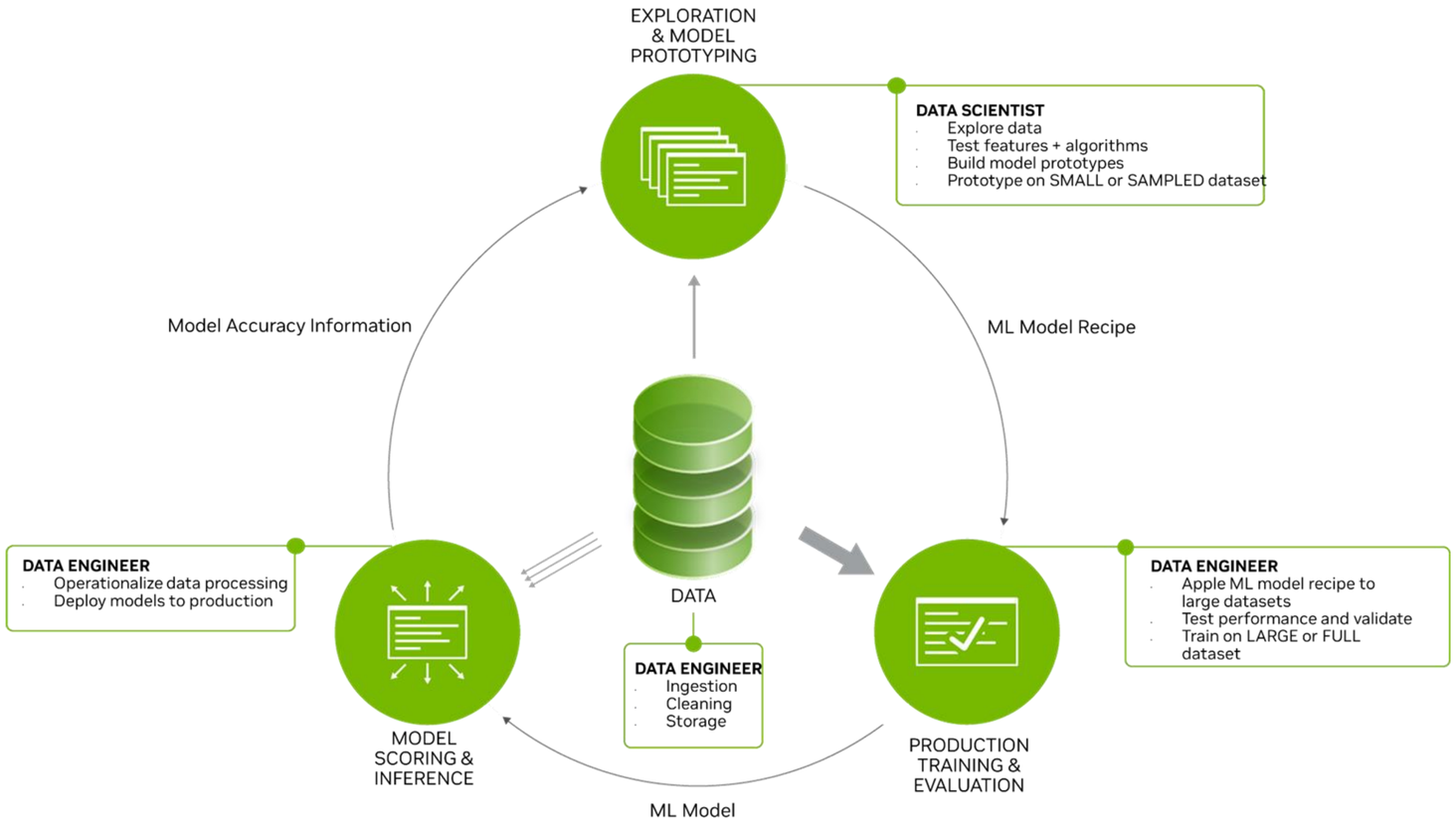
5x DGX-1 on InfiniBand network

Time in seconds — Shorter is better

cuIO / cuDF (Load and Data Preparation) Data Conversion XGBoost

Data Flywheel

Improve performance through the cyclical nature of data-driven growth and innovation



Hands-On Lab