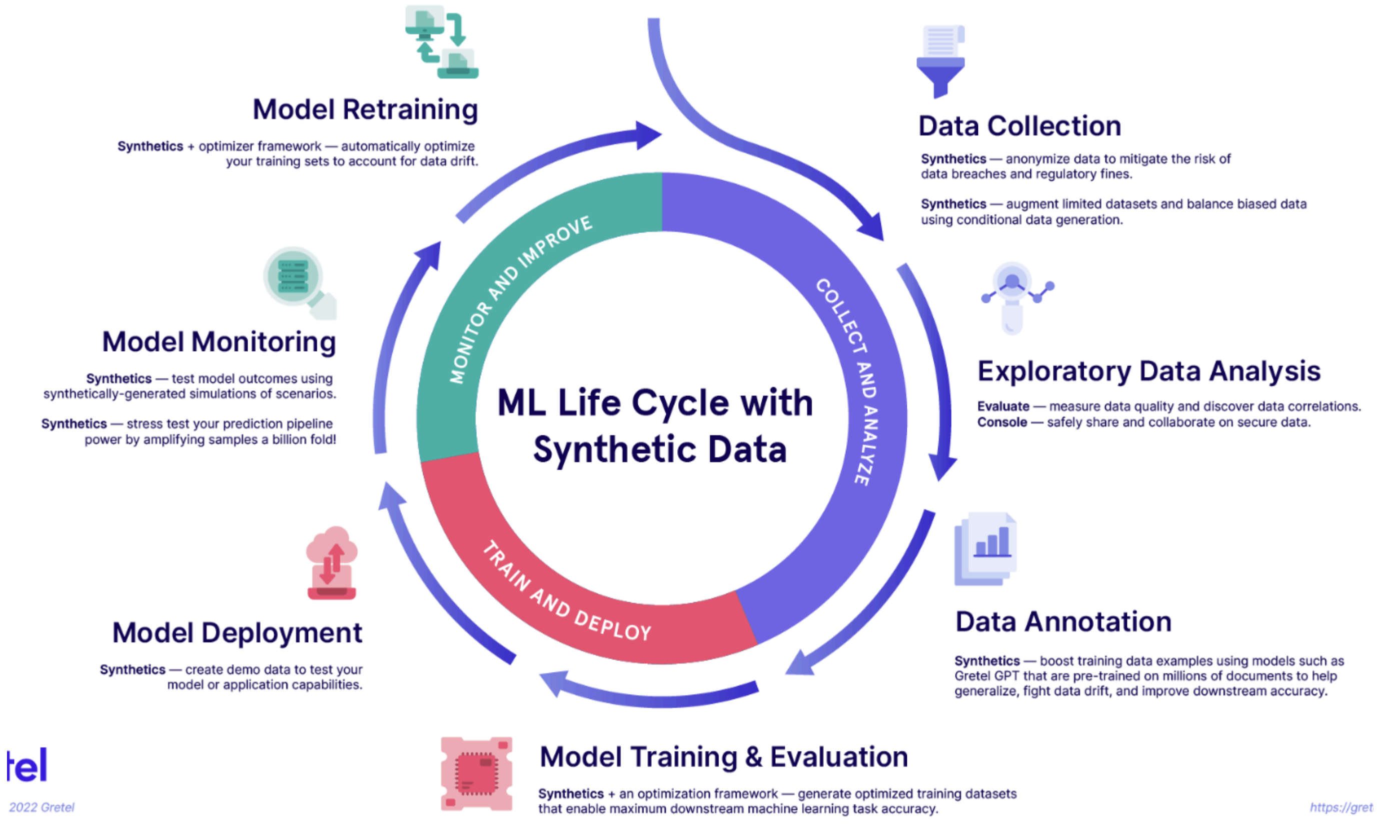
Machine learning life cycle

Machine learning life cycle is a process to build an efficient machine learning project. It is the process followed to develop, train, deploy, and maintain machine learning models. It’s focused on streamlining the stages necessary to develop machine learning models, deploy them to production, and maintain and monitor them. These steps are a collaborative process, often involving data scientists and DevOps engineers.



Data Collection
Synthetics — anonymize data to mitigate the risk of data breaches and regulatory fines.
Synthetics — augment limited datasets and balance biased data using conditional data generation.

Exploratory Data Analysis
Evaluate — measure data quality and discover data correlations.
Console — safely share and collaborate on secure data.

Data Annotation
Synthetics — boost training data examples using models such as Gretel GPT that are pre-trained on millions of documents to help generalize, fight data drift, and improve downstream accuracy.

Model Training & Evaluation
Synthetics + an optimization framework — generate optimized training datasets that enable maximum downstream machine learning task accuracy.

Model Deployment
Synthetics — create demo data to test your model or application capabilities.

Model Monitoring
Synthetics — test model outcomes using synthetically-generated simulations of scenarios.
Synthetics — stress test your prediction pipeline power by amplifying samples a billion fold!

Model Retraining
Synthetics + optimizer framework — automatically optimize your training sets to account for data drift.

In this document, we will be using the Cross-Industry Standard Process for the development of Machine Learning applications with **Quality assurance methodology (CRISP-ML(Q))** to explain each step in the machine learning life cycle.

The CRISP-ML(Q) is an industrial standard for building sustainable machine learning applications.

The 7 steps in a standard machine learning life cycle

1. Planning and identifying key business problems
2. Data collection
3. Exploratory data analysis
4. Data preparation and annotation
5. Feature Engineering
6. Model training, performance evaluation, and final selection
7. Model Deployment
8. Model monitoring
9. Updating and retraining models

1. Planning and identifying key business problems

Before jumping into any machine learning project, it is essential to clearly understand the business problem. This means identifying what the business is trying to achieve, what challenges exist in the current process, and whether those challenges can be addressed using machine learning. Not every problem needs an ML solution—sometimes traditional programming or automation is more suitable.

Once the business objective is understood, planning begins. This includes exploring different aspects of the problem and thinking through key questions:

- **Can machine learning solve this problem effectively?**
- **What kind of ML task fits the problem (e.g., classification, regression)?**
- **What does success look like for the business, and how will we measure it?**

During the planning phase, teams should assess the **feasibility** of the solution. This includes checking if enough quality **data is available**, whether it can be continuously updated, and if synthetic data can help reduce costs. Teams should also evaluate the **legal and ethical implications**, ensuring data is collected and used responsibly and lawfully. Planning also involves assessing the cost-benefit, preparing for a phased rollout, and making sure the solution is explainable, scalable, and robust in real-world conditions.

A final output of this phase is a detailed feasibility report. This document summarizes all key findings and decisions: data readiness, solution applicability, infrastructure needs, required skill sets, legal approvals, and measurable success metrics. This report acts as a roadmap for the development team and stakeholders, ensuring alignment across both technical and business perspectives.

2. Data Collection.

Once the business problem is clearly defined and planning is complete, the next crucial step is data collection. Machine learning models learn from data, so collecting the right type and amount of data is essential for building an effective solution. The quality of the model heavily depends on the quality of the data it is trained on.

In this phase, the goal is to gather all relevant data required to address the problem. This may include **historical records**, **user interactions**, **transaction logs**, **sensor data**, or even **external datasets** such as public databases or APIs. It's important to ensure that the data collected is representative of the real-world scenario the model is meant to handle.

Key considerations during this phase include:

- **Data sources**: Where is the data coming from? Is it internal (within the organization) or external?
- **Data volume**: Do we have enough data to train a reliable model?
- **Data variety**: Are we collecting all relevant features or types of data (structured, unstructured, images, text, etc.)?
- **Data freshness**: Is the data up-to-date? Will it continue to be updated in the future?
- **Data quality**: Is the data accurate, complete, and free from errors or duplicates?

Another important aspect is ensuring that **data collection follows ethical and legal standards**. Any personal or sensitive information must be handled carefully, with proper consent and compliance with privacy regulations such as GDPR or local data protection laws.

3. Exploratory data analysis

After collecting the data, the next critical step in a machine learning workflow is **Exploratory Data Analysis (EDA)**. EDA is the process of examining and understanding the dataset before building any models. It helps uncover patterns, detect anomalies, test assumptions, and gain insights into the structure and relationships within the data.

The goal of EDA is to answer important questions such as:

- What are the main characteristics of the dataset?
- Are there any missing or inconsistent values?
- What are the relationships between variables?
- Are there outliers or unusual trends?
- Does the data distribution match our assumptions?

During EDA, various techniques are used to explore the data:

- ➔ **Summary statistics**: Mean, median, standard deviation, minimum, and maximum values to understand data distribution.
- ➔ **Data visualization**: Charts and plots such as histograms, box plots, scatter plots, and correlation heatmaps help to visually interpret the data.
- ➔ **Missing value analysis**: Identifying where and how much data is missing.
- ➔ **Outlier detection**: Finding and understanding extreme values that may affect model performance.
- ➔ **Feature relationships**: Checking correlations and patterns between variables to guide feature selection and engineering.

EDA not only helps in understanding the data but also influences how the data should be cleaned, transformed, and modeled.

4. Data preparation and annotation

This stage is about cleaning, transforming, and labeling the data to make it ready for training a machine learning model. Well-prepared data improves model accuracy and ensures reliable results.

Data preparation involves several tasks:

- **Handling missing values**: Filling in, removing, or imputing missing data using techniques like mean substitution or predictive models.
- **Removing duplicates**: Eliminating repeated records that can bias the model.
- **Outlier treatment**: Removing or adjusting extreme values that may affect training.
- **Data transformation**: Normalizing or standardizing data so all features are on a similar scale.
- **Encoding categorical data**: Converting text-based categories (like "Male" or "Female") into numeric formats using techniques like one-hot encoding or label encoding.
- **Feature selection and extraction**: Identifying the most relevant features or creating new ones from the existing data to improve model performance.

Data annotation is especially important for supervised machine learning tasks. It refers to labeling the data with the correct answers. For example, in a spam detection model, emails need to be labeled as "spam" or "not spam". In computer vision, images must be labeled with the correct objects or bounding boxes. Annotation can be done manually by experts, semi-automatically using rule-based systems, or even using pre-labeled datasets.

5. Feature Engineering

Feature Engineering is the process of selecting, modifying, or creating new features (input variables) from raw data to improve the performance of a machine learning model. Good features can make a simple model perform surprisingly well, while poor features can limit even the most advanced algorithms.

The goal of feature engineering is to **highlight the most meaningful aspects of the data** that help the model understand the patterns behind the target output.

Some common steps in feature engineering include:

- **Feature selection**: Choosing only the most relevant features for the model and removing irrelevant or redundant ones. This helps reduce complexity and improves efficiency.
- **Feature transformation**: Converting features into formats that are more suitable for the model. For example, applying log transformations to skewed data or normalizing numerical features so they're all on the same scale.
- **Encoding categorical variables**: Converting text-based data into numbers using techniques like one-hot encoding, label encoding, or ordinal encoding.
- **Creating new features**: Sometimes, combining or modifying existing features can create more useful insights. For example, if you have `Date of Birth`, you can derive a new feature called `Age`.
- **Handling time-based features**: For time-series data, extracting day, month, year, or trends (like moving averages) can help capture patterns over time.
- **Dimensionality reduction**: When working with high-dimensional data, techniques like PCA (Principal Component Analysis) are used to reduce the number of features while retaining important information.

Good feature engineering often requires both domain knowledge and a deep understanding of the data. It bridges the gap between raw data and meaningful machine learning insights.

6. Model training, performance evaluation, and final selection

After the data has been cleaned, prepared, and the right features have been engineered, the next major step in the machine learning pipeline is **model training**. This is the phase where the machine learning algorithm learns patterns from the data and builds a predictive model.

i) Model Training

Model training involves feeding the training dataset (features and their corresponding labels) into a machine learning algorithm. The algorithm then **learns the relationships** between input features and the target output. Depending on the nature of the problem, different types of models may be used. such as linear regression, decision trees, random forests, support vector machines, or neural networks.

The dataset is usually divided into:

- **Training set** – used to train the model.
- **Validation set** – used to fine-tune parameters and prevent overfitting.
- **Test set** – used to evaluate how well the final model performs on unseen data.

ii) Performance Evaluation

After training, the model's performance needs to be measured using appropriate evaluation metrics. The choice of metric depends on the type of problem (classification, regression, etc.):

- For **classification problems**: Accuracy, Precision, Recall, F1-score, ROC-AUC.
- For **regression problems**: Mean Absolute Error (MAE), Mean Squared Error (MSE), R-squared (R^2).

Evaluation also involves checking for:

- **Overfitting**: When the model performs well on training data but poorly on new data.
- **Underfitting**: When the model fails to capture patterns even in training data.

iii) Final Model Selection

After evaluating multiple models or configurations, the best-performing model is selected based on:

- **Model performance metrics**
- **Generalization ability (low overfitting)**
- **Interpretability and explainability**
- **Training time and computational cost**
- **Scalability and deployment feasibility**

In many cases, **hyperparameter tuning** is also performed to improve the model's performance further. This involves adjusting settings like learning rate, depth of trees, number of neurons, etc., using tools like Grid Search or Random Search.

7. Model Deployment

Once a machine learning model is trained, evaluated, and selected, the next critical step is Model Deployment. This is the process of integrating the trained model into a real-world application where it can make predictions on live or unseen data. Model deployment marks the point where data science meets real-world impact. Once deployed, the model must also be **monitored** to ensure it continues to perform as expected.

8. Model monitoring

It's crucial to continuously monitor the model's performance to ensure it remains accurate, reliable, and useful over time. This process is known as Model Monitoring.

In real-world environments, data can change frequently—a phenomenon called data drift or concept drift—which can lead to a decline in model performance. Monitoring helps detect such issues early and take corrective actions.

What Should Be Monitored?

1. **Model Performance Metrics**: Regularly track metrics like accuracy, precision, recall, F1 score, or RMSE on live data. A sudden drop may indicate that the model is no longer valid for the new data patterns.
2. **Data Drift**: Detects changes in the distribution of input data compared to the training data. For example, if a feature used in predictions starts showing different behavior, the model might produce unreliable outputs.
3. **Prediction Distribution**: Monitor how the model's predictions are distributed. A shift in the types of predictions being made (e.g., class imbalance) could be a red flag.
4. **Input Anomalies**: Check for unexpected values, missing data, or data quality issues in real-time inputs. Bad data can directly affect model accuracy.
5. **Latency and Throughput**: Measure how long the model takes to respond to prediction requests and how many predictions it can handle. This ensures the system remains responsive and scalable.
6. **User Feedback and Business KPIs**

8. Updating and retraining models

Machine learning models are not a “train once and forget” solution. Over time, the real-world data your model sees will evolve, and the model may begin to perform poorly. This is where model updating and retraining come into play.

Retraining is the process of taking new or updated data, feeding it back into the model pipeline, and building an improved version of the model that reflects the latest trends and patterns.