VLAIO TETRA : MLOps4ECM

MLOps: Why and how?

Met steun van VLAIO

HBC.2023.0062

In 2022 only 26% of organizations had AI projects deployed in production.
– O'Reilly

Why?

Are you already working with Machine Learning in your company context?

If yes, which hurdles did/do you experience to bring these models in production?

Why?

Poor communication

Lack of versioning

Lack of automation

Dynamic properties of AI environments

Poor cooperation

Unclear and unrealistic expectations

And more ...

The solution is MLOps

A Paradigm used to **create** and **maintain** a machine learning model that will be deployed in **production**

DevOps

- Past: Development and Operations = 2 people → separate

Developer: builds application

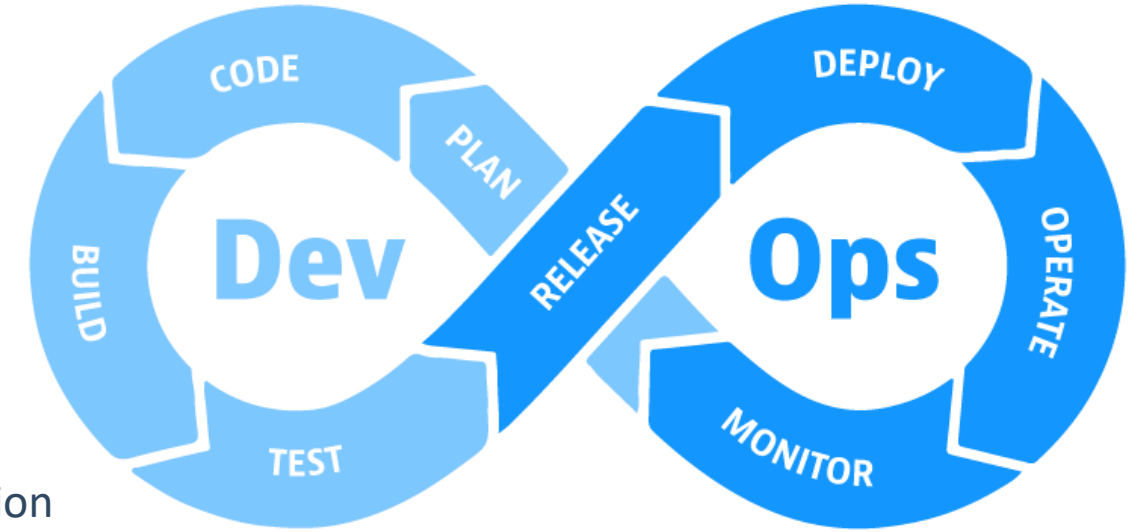
Wall of confusion

Operations: deploys application

DevOps

- Present

Developer: builds application



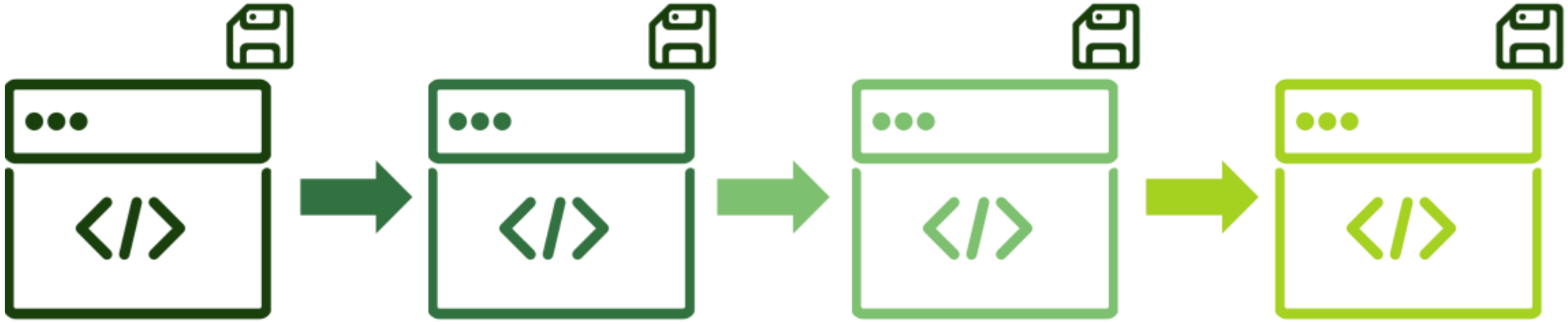
Operations: deploys application

DevOps principles: Collaboration

- Planning with **everyone** in the room
- **Communication** between parties
- Short feedback loops with **customer**

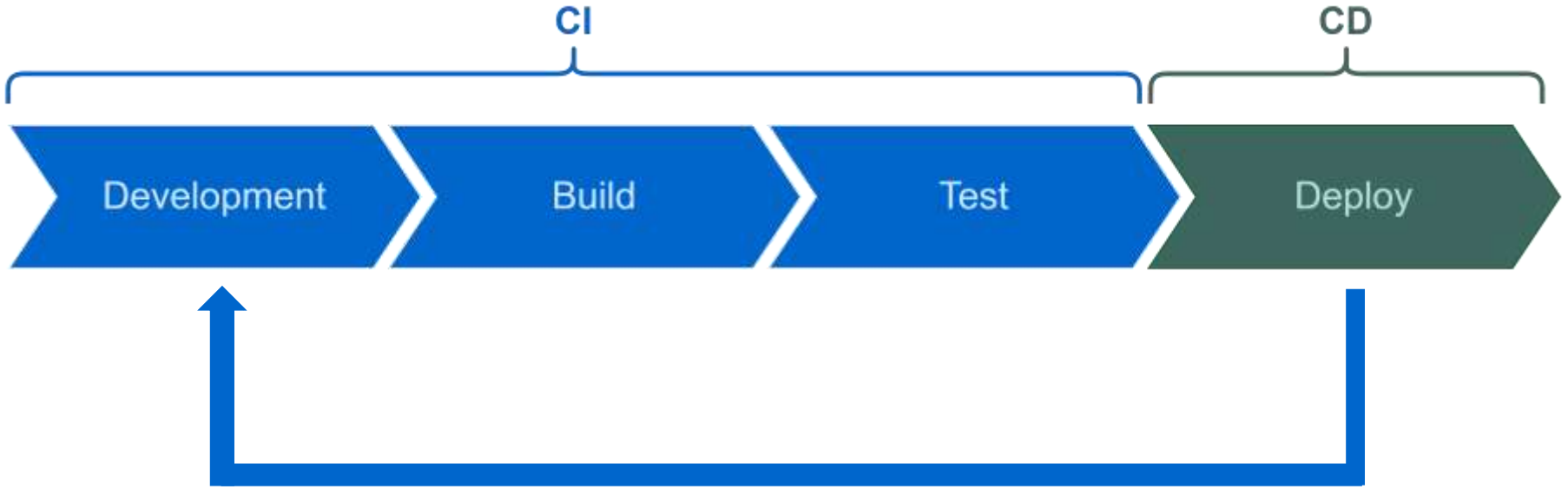
DevOps principles: Reproducibility

- Code versioning



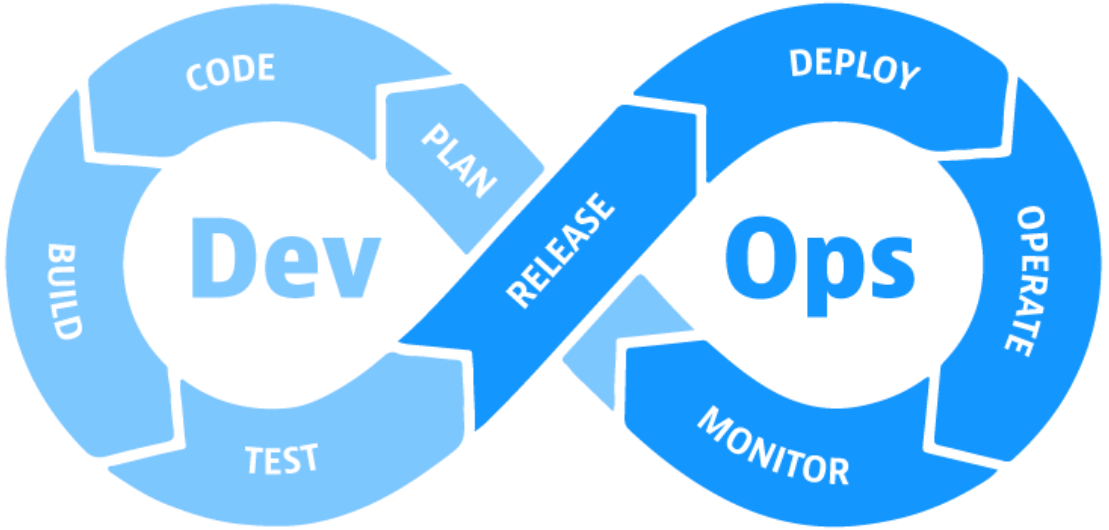
DevOps principles: Continuous improvement

- Continuous integration and continuous deployment automation (CI/CD)
- Feedback loops



DevOps principles: Automation

- Automatic testing
- CI and CD happen automatically
- Use of tools : e.g. github actions



GitHub Actions

DevOps principles: Monitoring

- Overseeing the entire process
- Integration and testing overview
- Status of all aspects in the production environment
- Use dashboards

MLOps

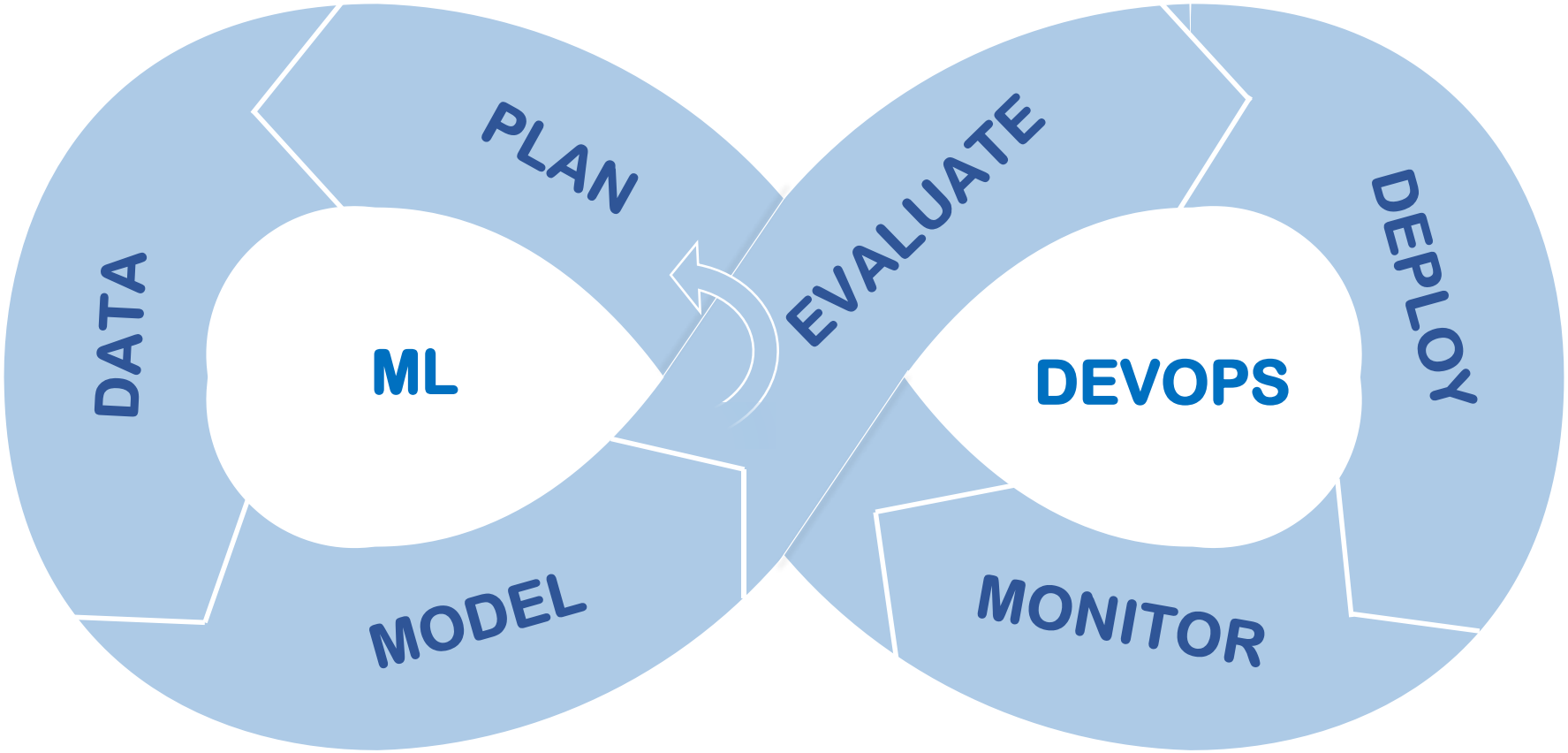
Machine Learning Application

=

DATA + MODEL + CODE

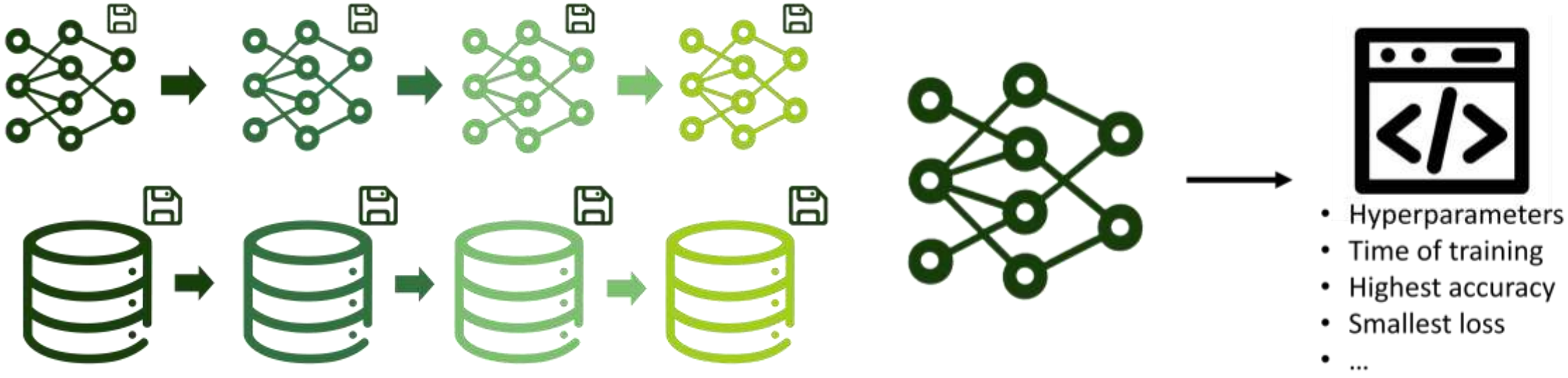
➔ Need for extra well-defined structures, processes, and proper software tools that manage these artifacts over the machine learning cycle

MLOps: pipeline



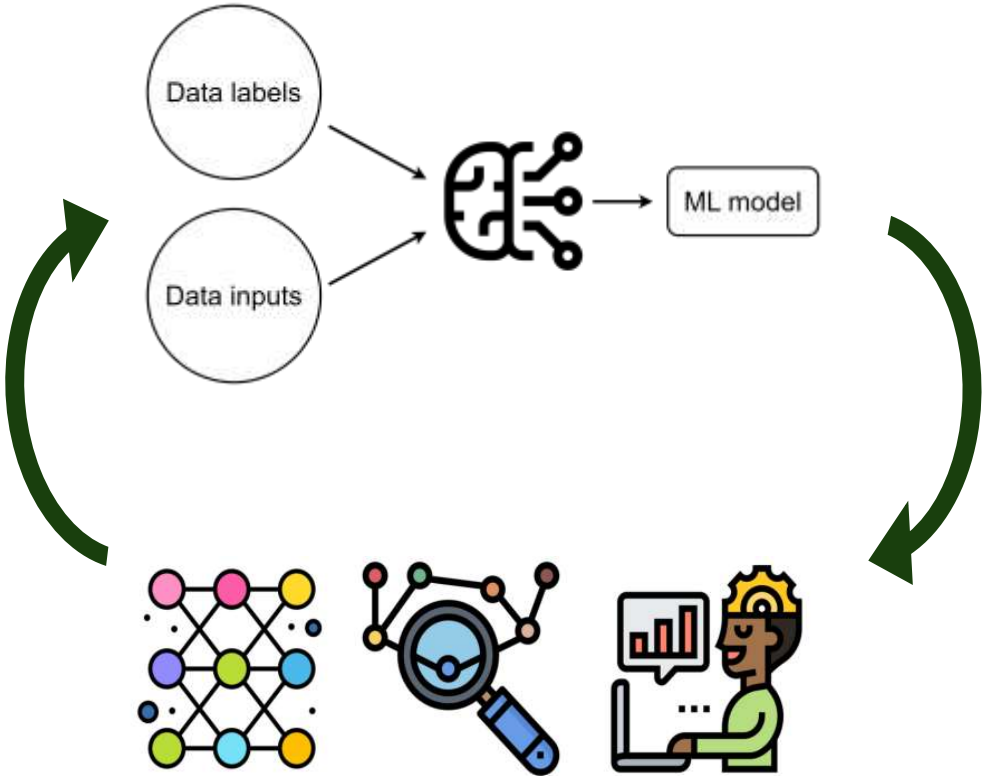
MLOps principles: Reproducibility

- Versioning code, **data and model**
- Tracking **metadata and logging**



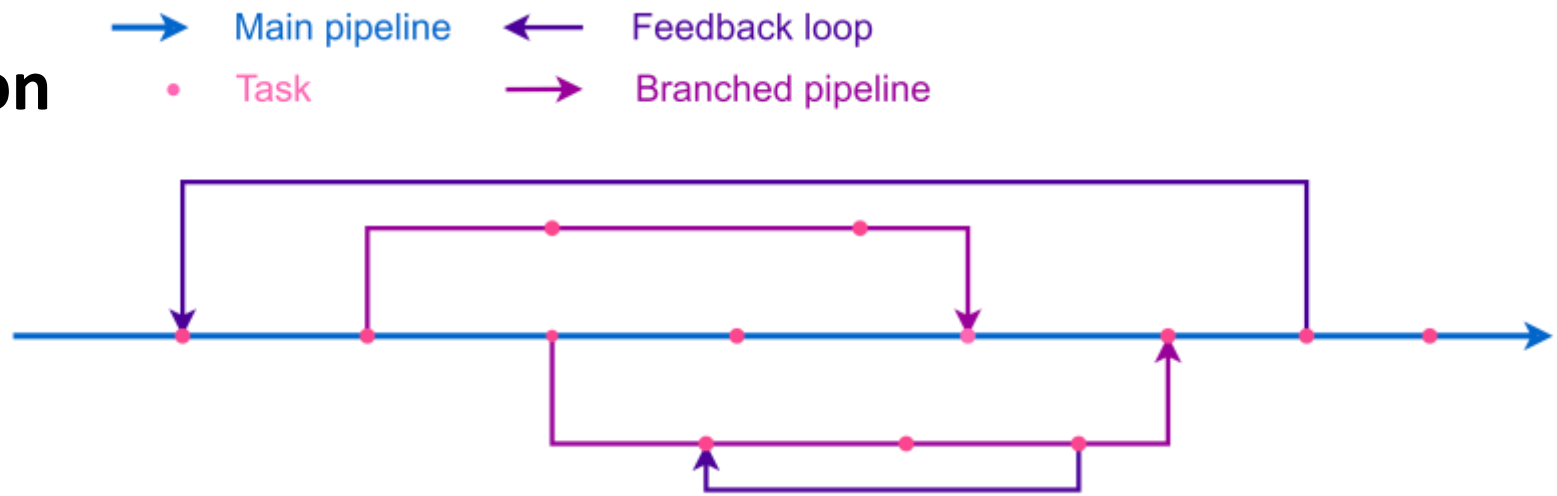
MLOps principles: Continuous improvement

- CI/CD
- Feedback loops
- **Continuous model Learning (CL)**
- **Continuous model evaluation**



MLOps principles: Automation

- **Automatic model training**
- Automatic testing
- CI and CD happen automatically
- Use of tools
- **Workflow orchestration**



MLOps principles: Monitoring

- Overseeing the entire process
- Integration and testing overview
- Status of all aspects in the production environment
- **Monitor model inputs and model performance**
- Use dashboards

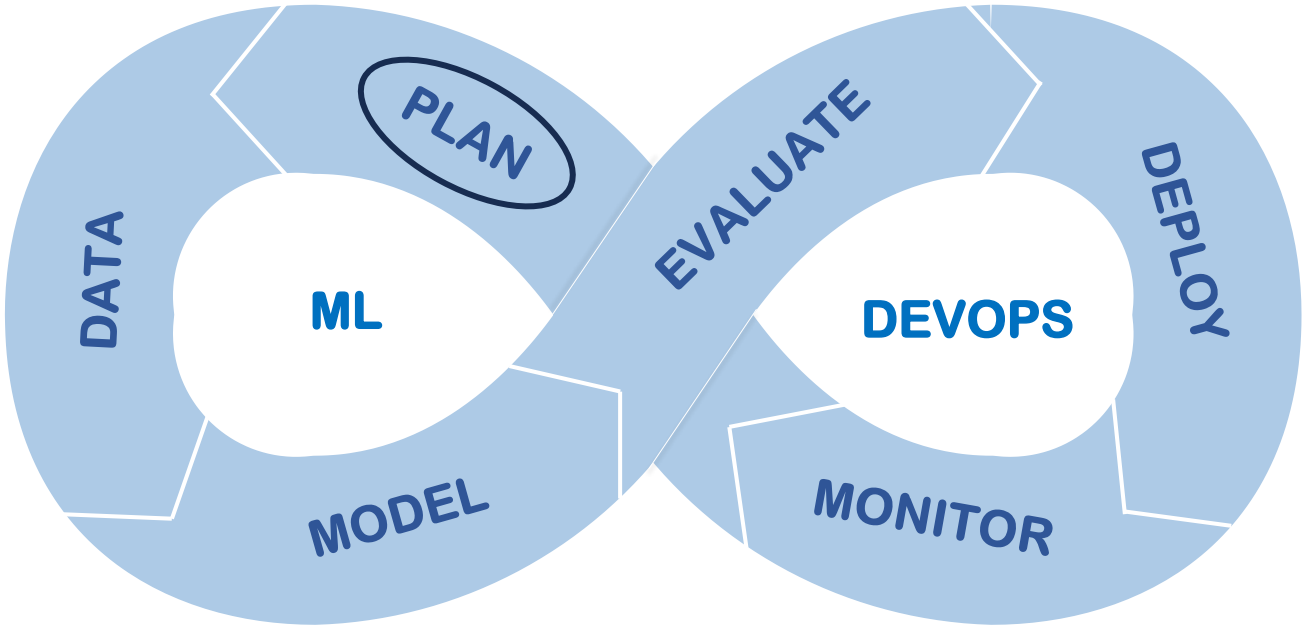
DevOps vs. MLOps

	DevOps	MLOps
Code versioning	✅	✅
Compute environment	✅	✅
Continuous integration/delivery (CI/CD)	✅	✅
Monitoring in production	✅	✅
Data provenance		✅
Datasets		✅
Models		✅
Hyperparameters		✅
Metrics		✅
Workflows		✅

A Lot of tools …

The pipeline

- **Plan**
- Data
- Model
- Evaluate
- Deploy
- Monitor
- Closing the loop

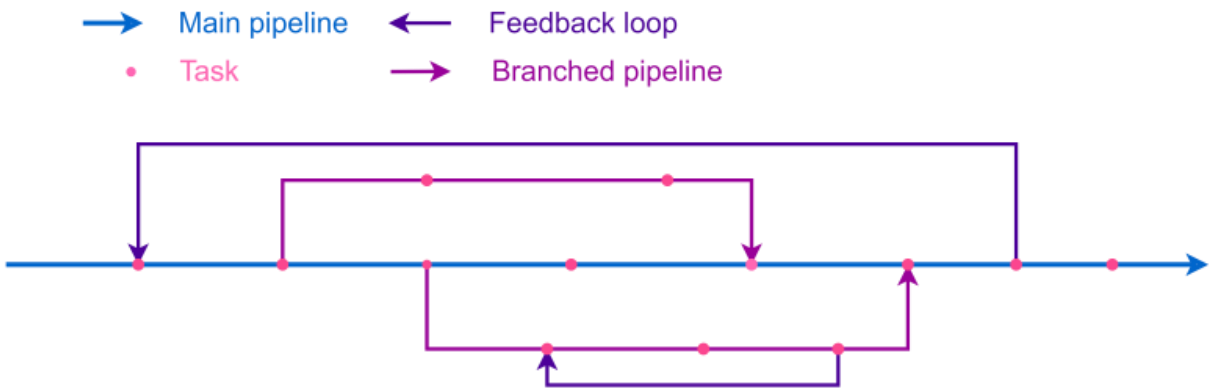


Plan – what

- Identify the problem together with all parties
- Plan a solution:
  - What does the solution need to do?
  - Which data needs to be collected?
  - Which type of model will be used?
  - Establish a workflow orchestration platform
  - Setup code version control
  - Decide on the used tools
  - …

Plan - workflow orchestration

- Create workflows and tasks
- Run the workflows + overview of these runs
  - Did run fully execute?
  - Where did an error occur?
  - …
- Scheduling, alerting …

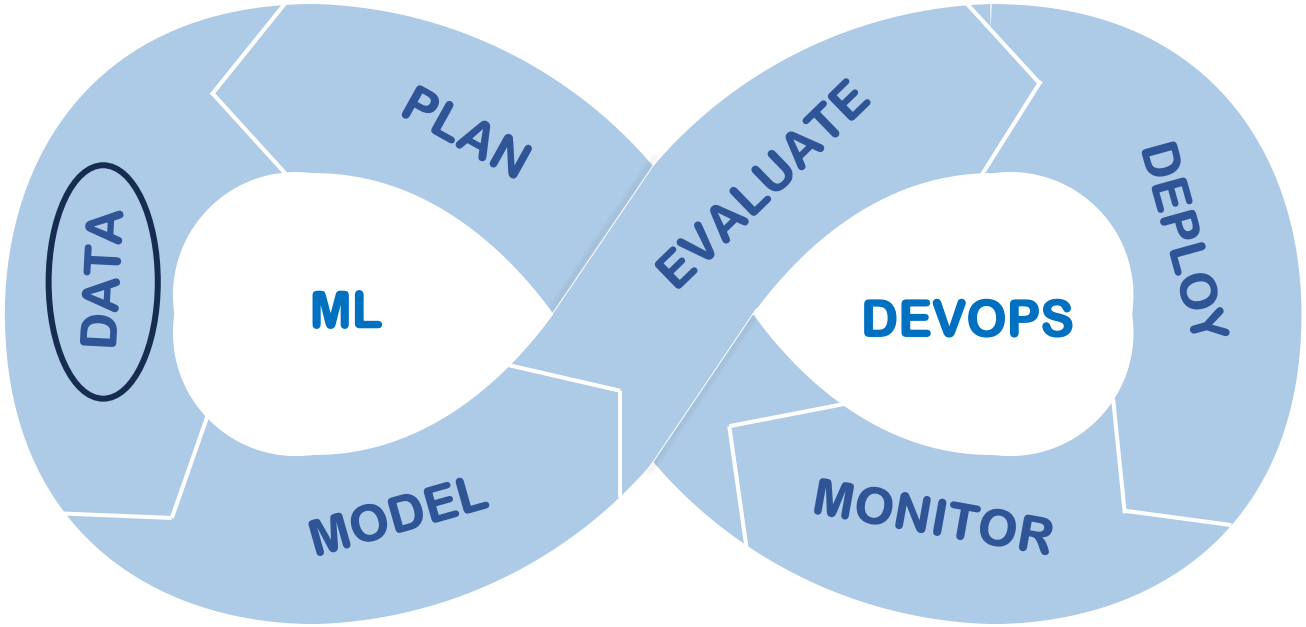


Plan – tools

- Version control: Github, Gitlab, …
- Workflow orchestration (pipeline tools): Prefect, Airflow, Argo …

The pipeline

- Plan
- **Data**
- Model
- Evaluate
- Deploy
- Monitor
- Closing the loop



Data – what

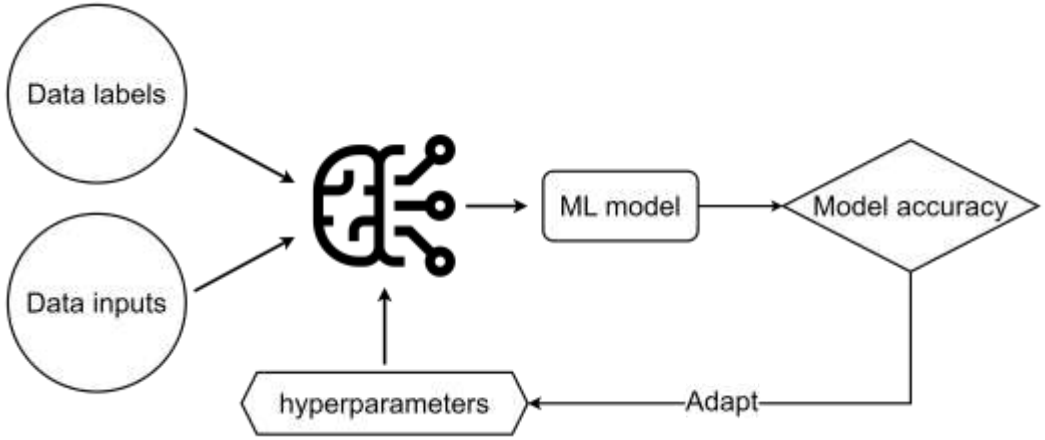
- Collect the data
- Data labelling
- Data preprocessing
- Create data versioning

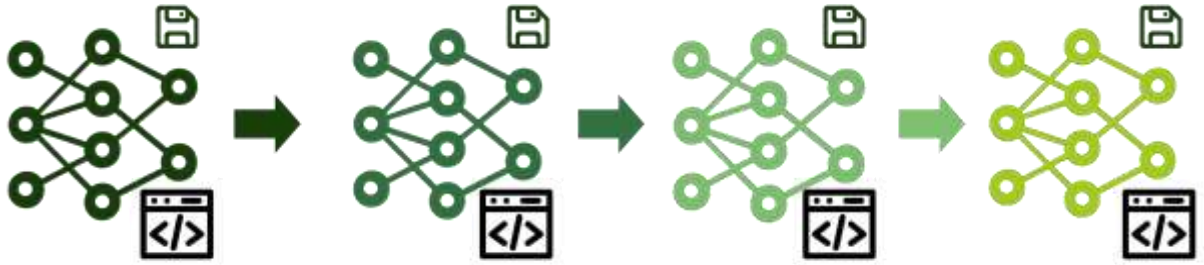
Data exploration/preprocessing tools

- Processing: Pandas , Python notebooks
- Visualisation: matplotlib, plotly, …
- Storage: Databases
- Labeling: labelstudio, labelbox, …
- Dashboarding: Plotly dash, streamlit, …

Model – what

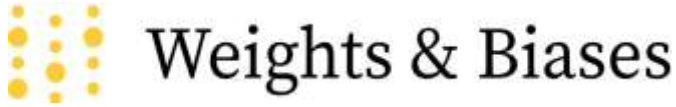
- Model architecture creation
- Model training
- Model finetuning
- Create model + metadata versioning





Model – Hyperparamter tuning

- AutoML → automated ML hyperparameter tuning
- Optuna, weights and biases, edge impulse…

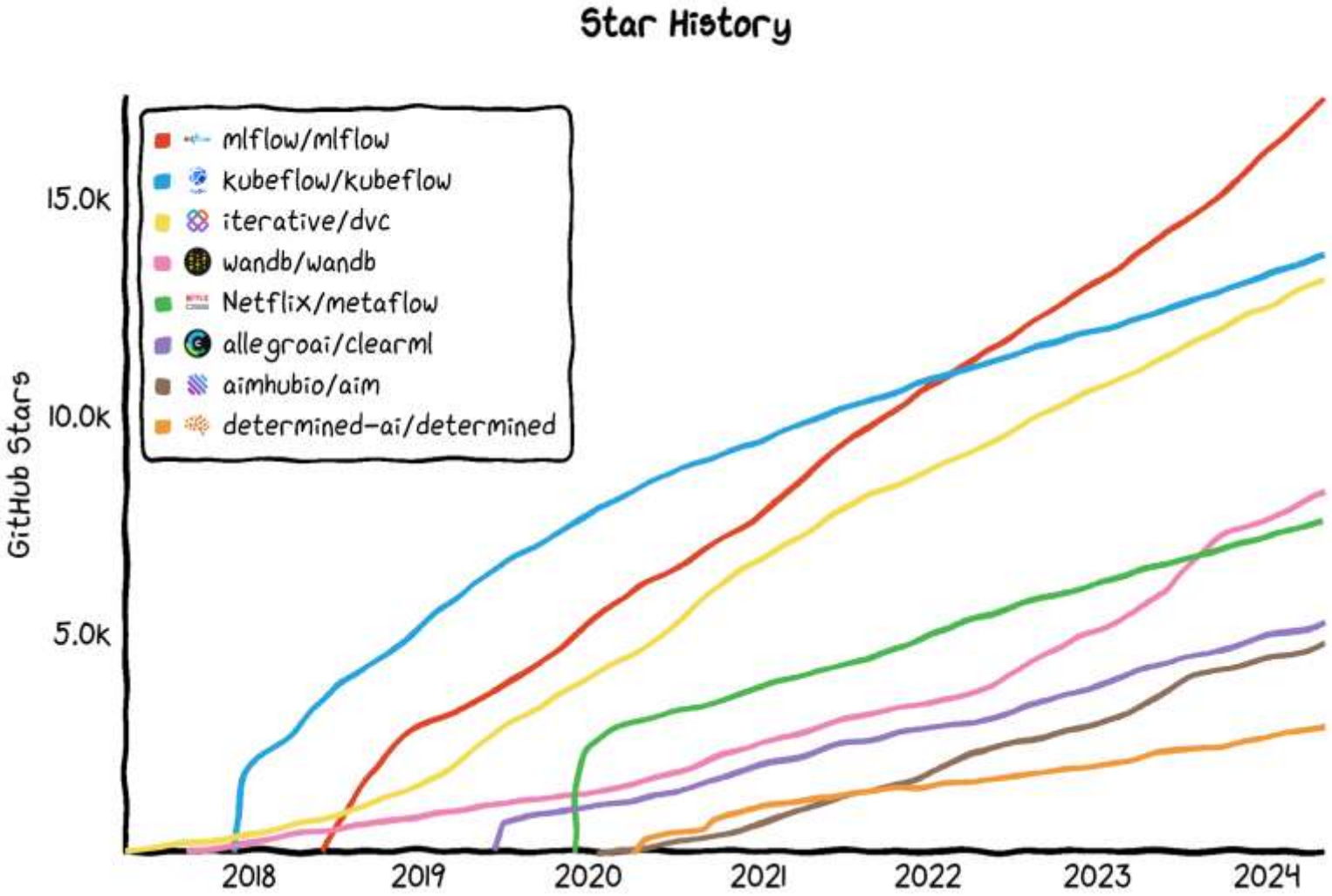


Models – tool

- Experiment trackir
- MLflow, Weights a

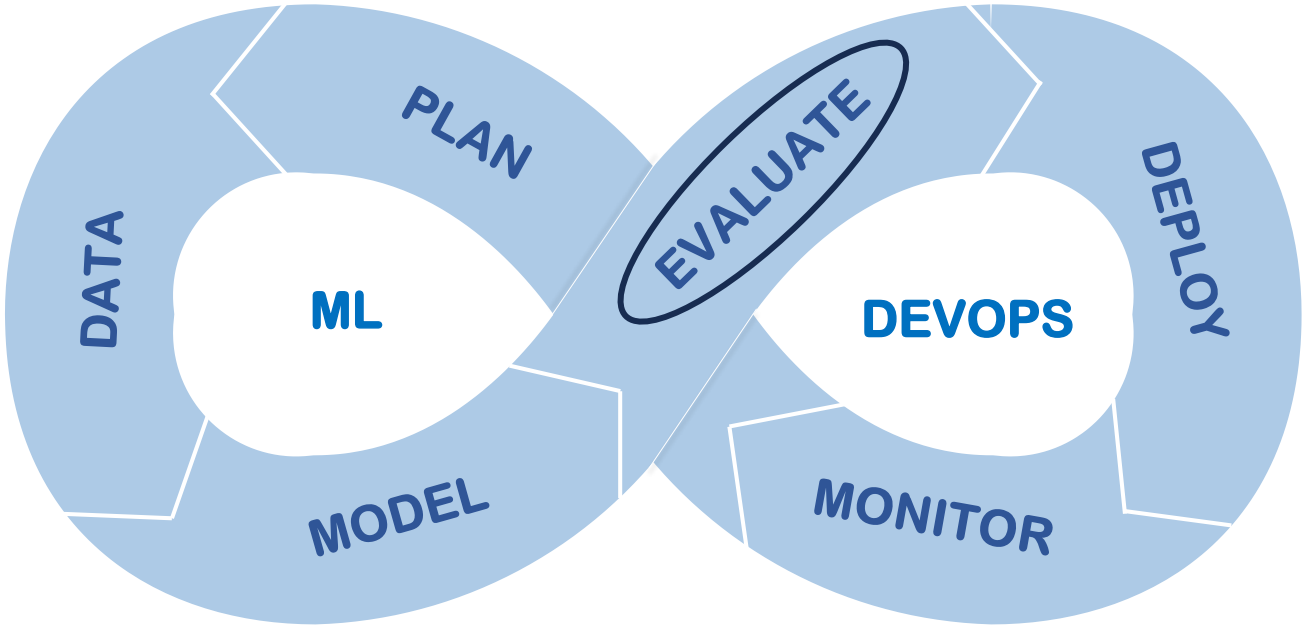
Weights & Biase

mlflow



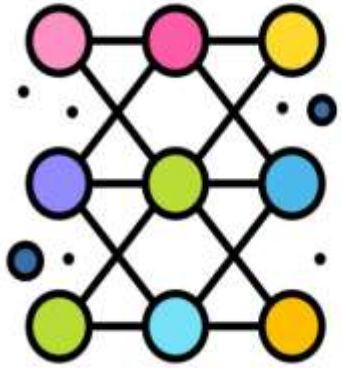
The pipeline

- Plan
- Data
- Model
- **Evaluate**
- Deploy
- Monitor
- Closing the loop



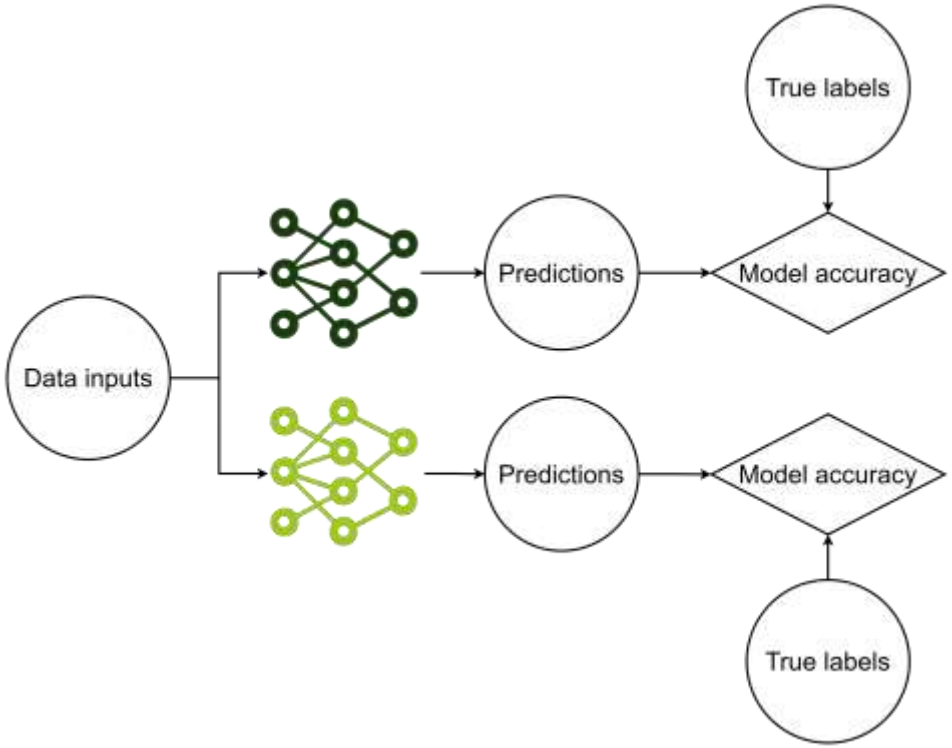
Evaluate – what

- Evaluate the created machine learning model in context
  - Choose metrics: accuracy, resource usage, fairness, business KPIs ...
- If there is already a model deployed
  - Champion/challenger testing
  - A/B testing



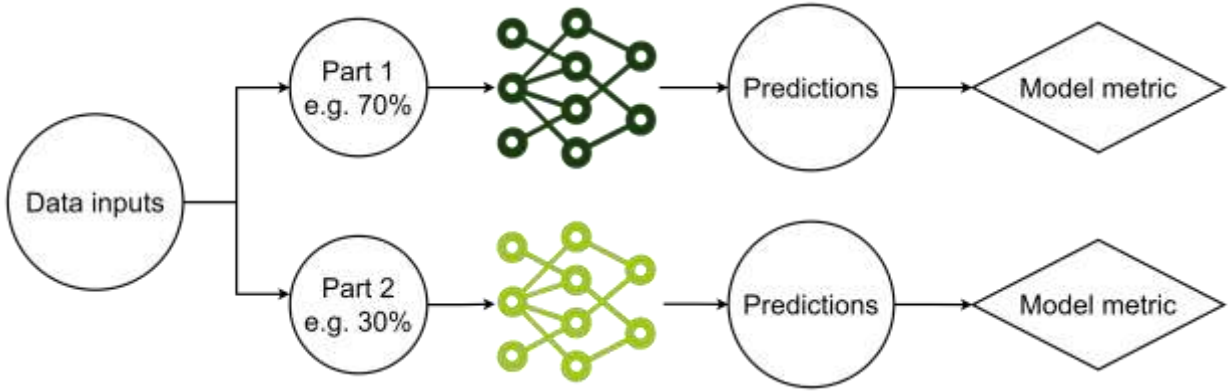
Champion testing (shadow testing)

- 1 model is currently deployed = champion
- Deploy **one/several** additional models = challengers
- **All models** receive and **score** the incoming requests but **only result champion is used**
- Log model metrics of all models
- When a challenger performs better → selected as **new champion**



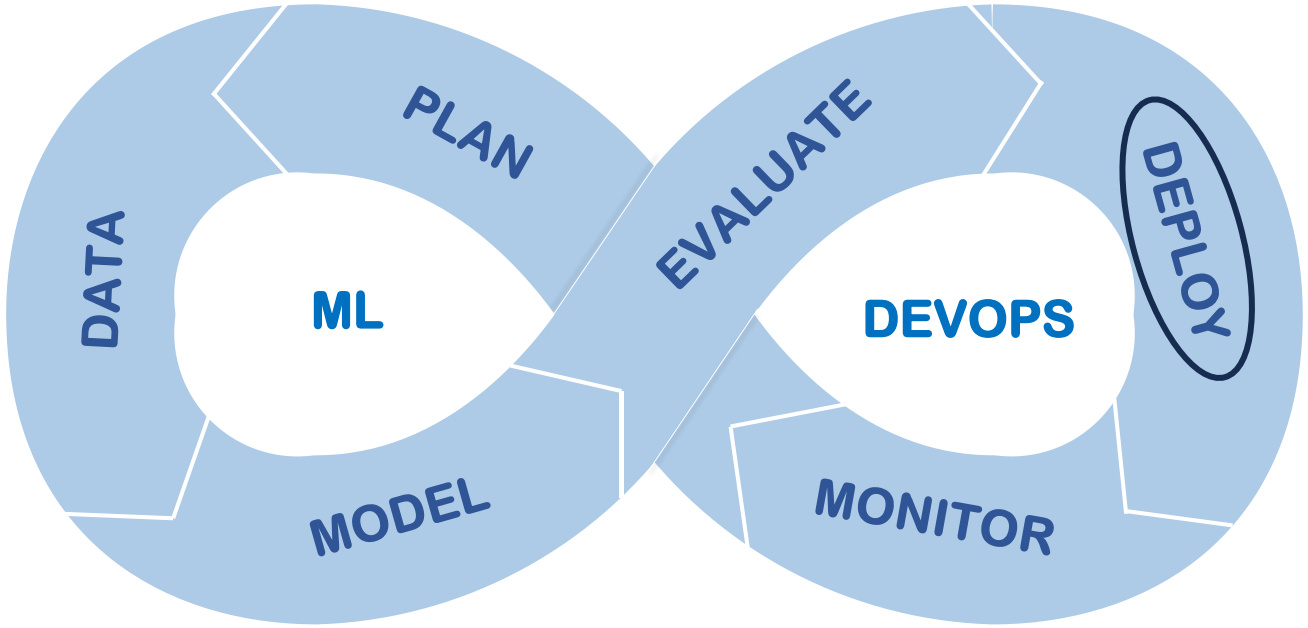
A/B testing

- If **no ground truth** or other metric than accuracy
- Deploy **only one** additional model
- Decide on a statistical metric
- Let the additional model **process part of the real data**
- Check if the additional model performed better



The pipeline

- Plan
- Data
- Model
- Evaluate
- **Deploy**
- Monitor
- Closing the loop



Deploy – what

- Prepare for production:
  - Model compression (vb. Quantization, pruning, model distillation)
  - Change to correct format (vb. ONNX)
  - Risk mitigation
- Create CI/CD pipelines
- Containerization = Docker

Deploy – tools

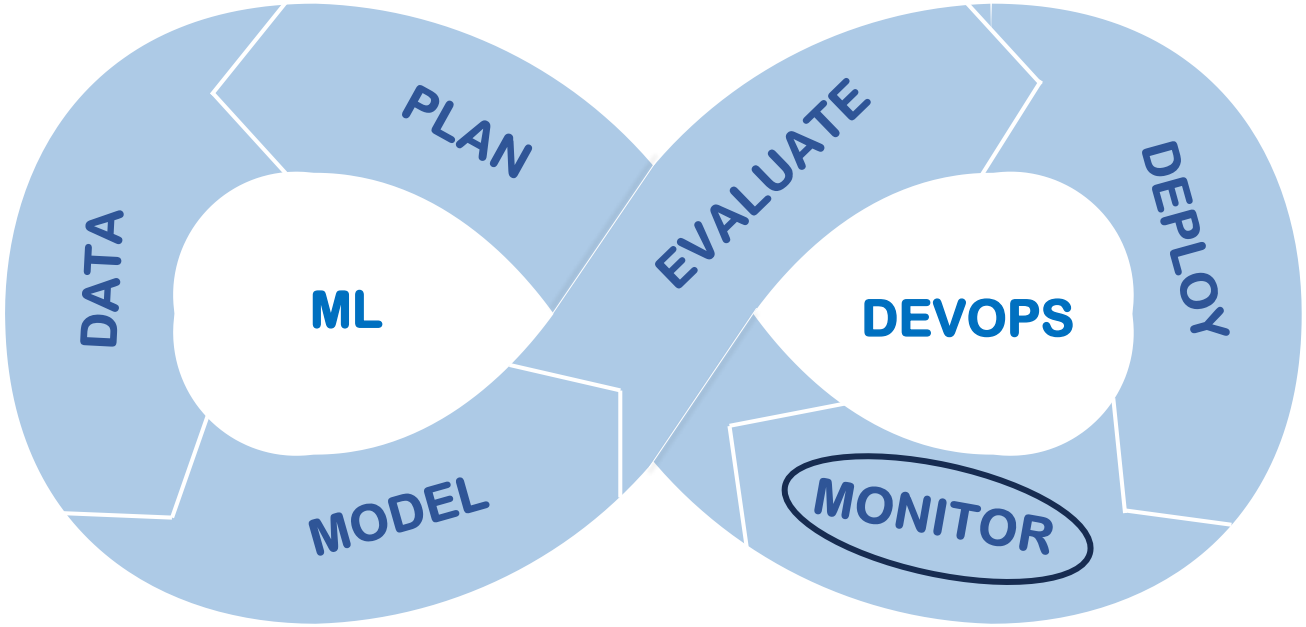
- Containerization: Docker, Kubernetes
- Deployment tools: Seldon.io, TensorFlow serving, Kubeflow, AWS Sagemaker, BentoML …

Amazon SageMaker

Kubeflow

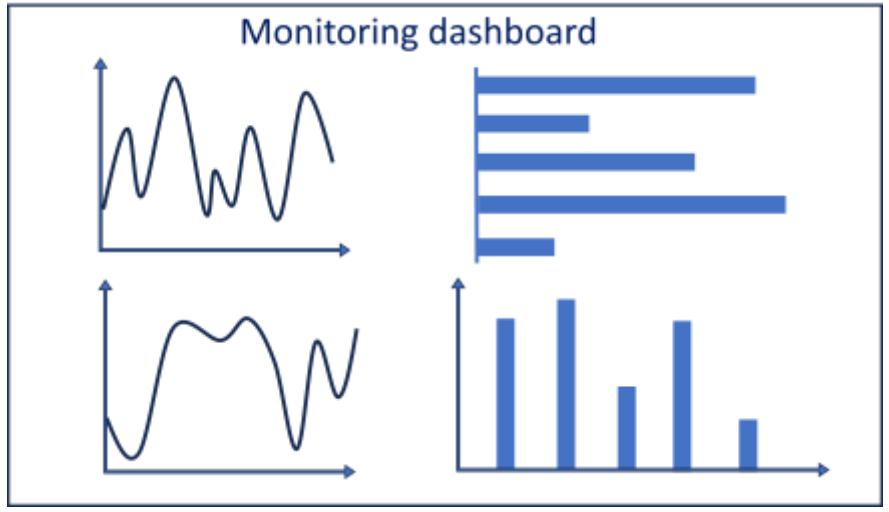
The pipeline

- Plan
- Data
- Model
- Evaluate
- Deploy
- **Monitor**
- Closing the loop



Monitor – what

- Monitor application for errors
- Monitor the data for data quality and data drift
- Monitor the machine learning model for concept drift
- Monitor other parameters for drift e.g. resources, fairness …

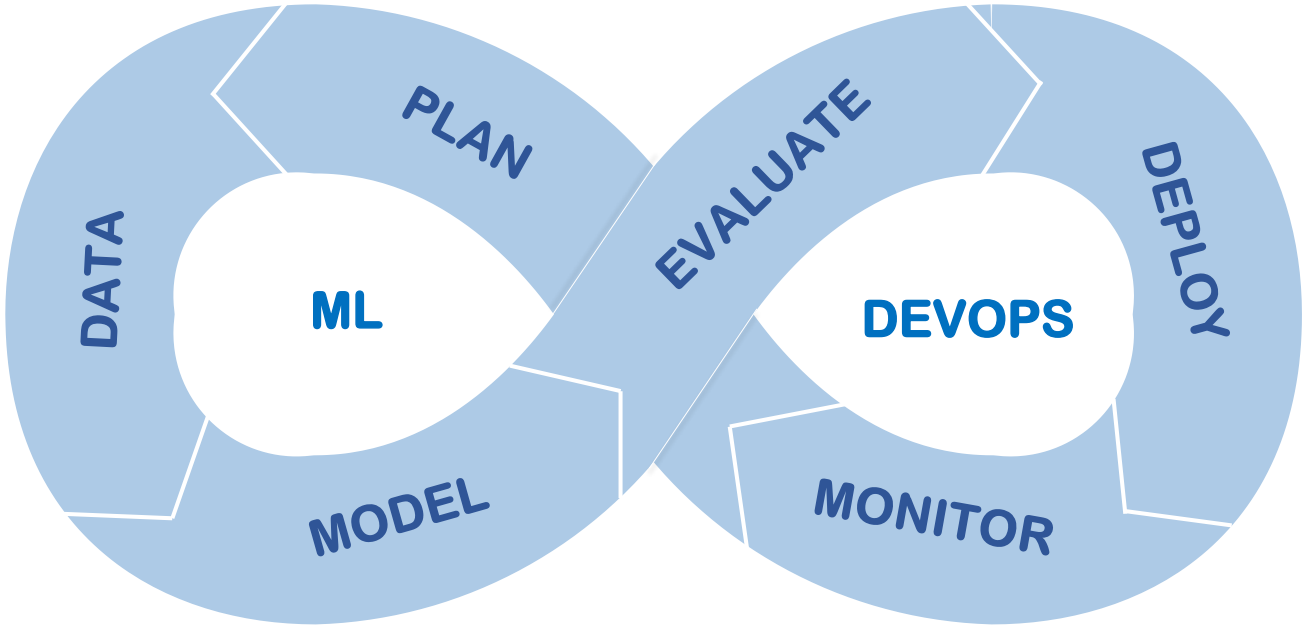


Monitoring – tools

- Tools that monitor input data, predictions and model quality

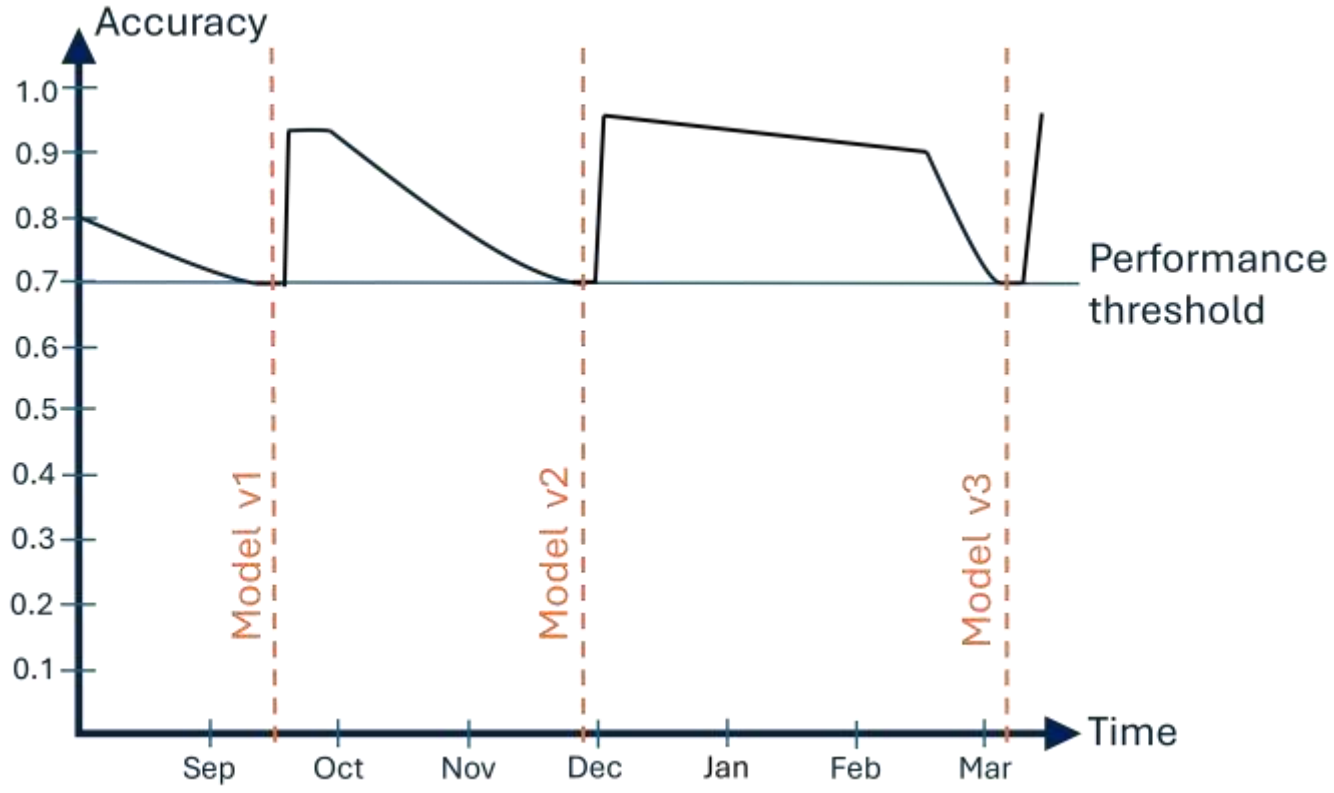
The pipeline

- Plan
- Data
- Model
- Evaluate
- Deploy
- Monitor
- **Closing the loop**



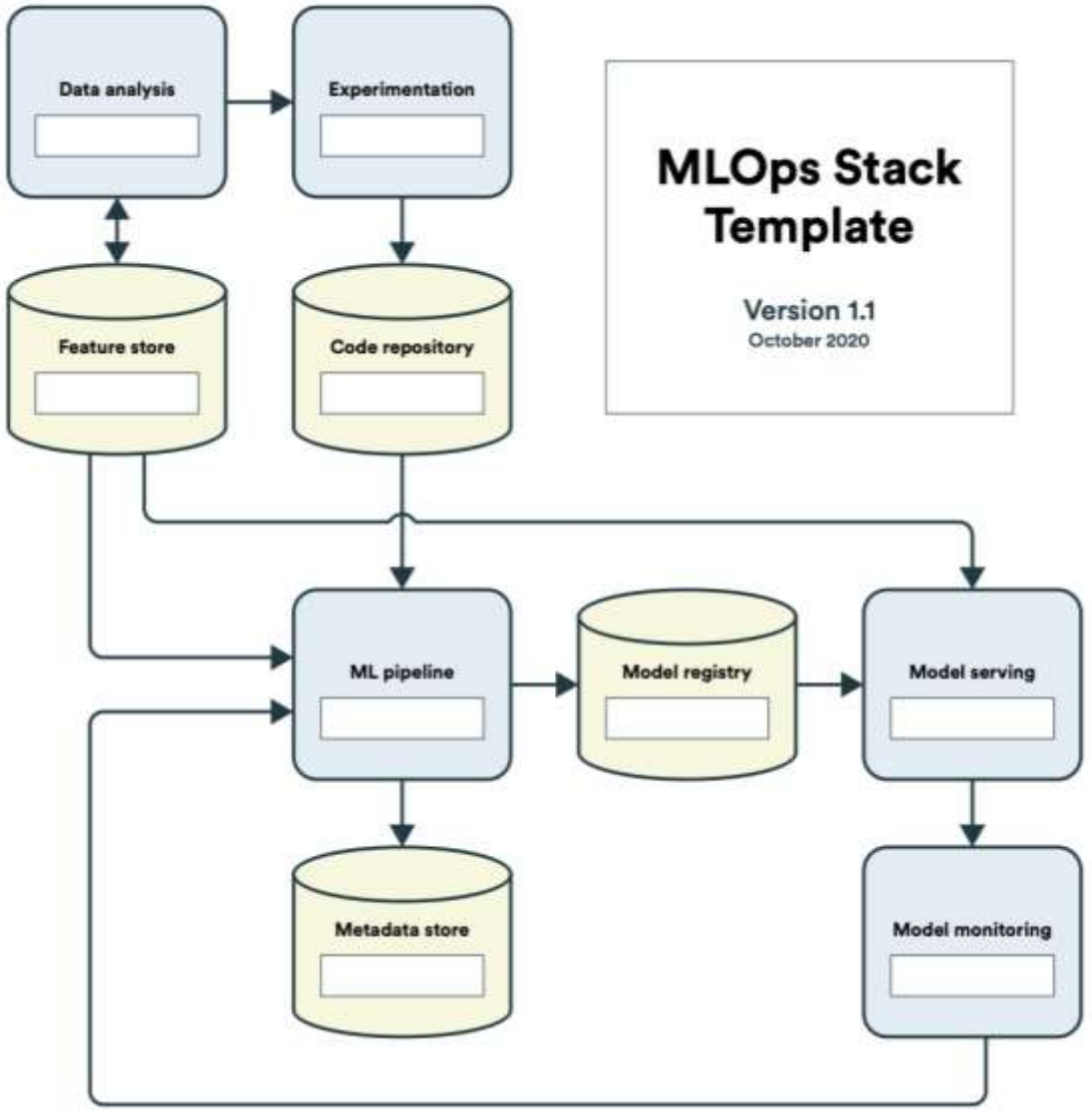
Closing the loop – what?

- Adapt your machine learning model
- Start the process again



MLOps stack

- Scope of each tool might span several components of the MLOps process
  - ➔Requires careful consideration depending in use case requirements
- MLOps stack template as guideline
- As for regular software development: thorough requirements analysis upfront!



VLAIO TETRA : MLOps4ECM

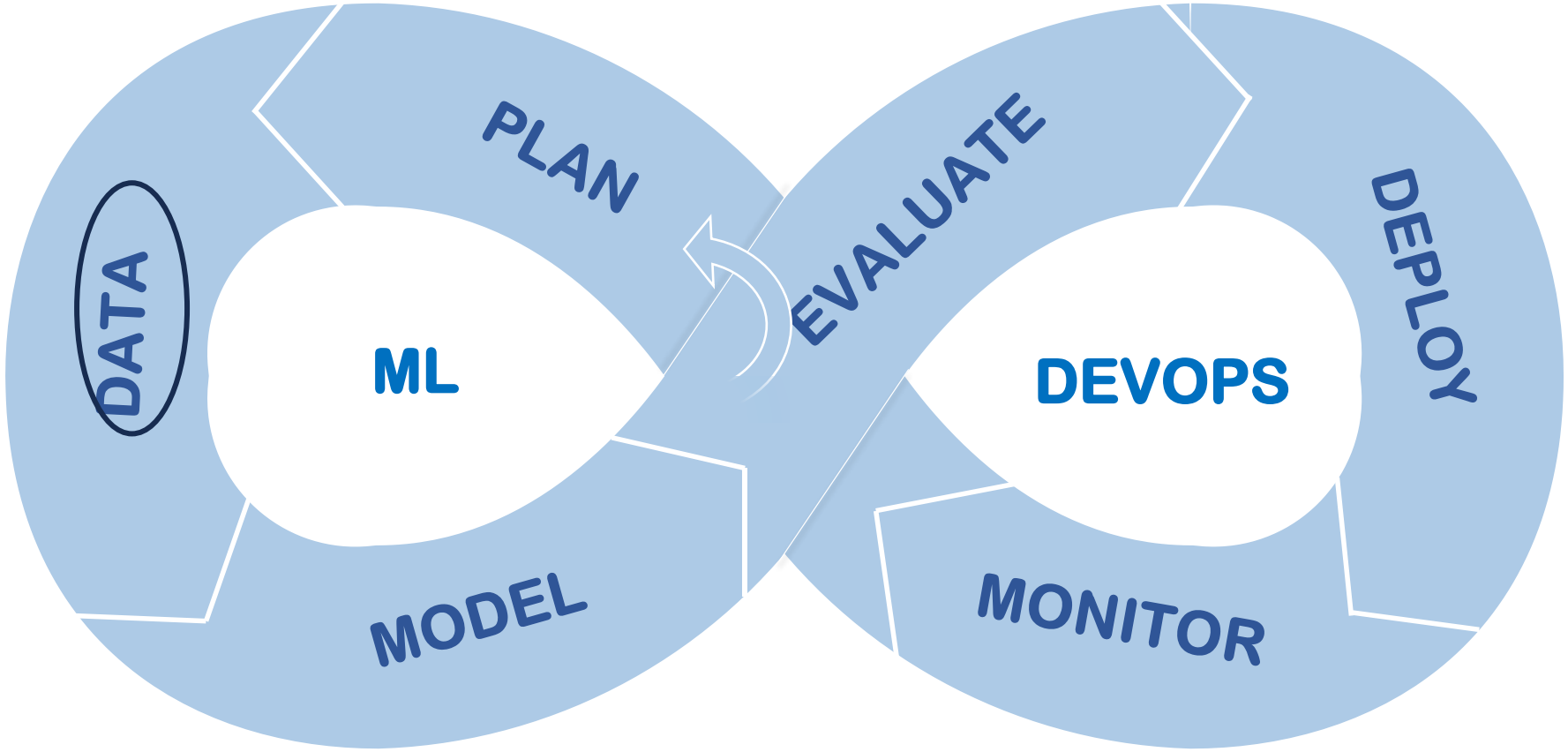
MLOps: Why and how?

Lara Luys

Met steun van VLAIO

HBC.2023.0062

MLOps: pipeline



What?

- Collect the data
- Data labelling
- Data preprocessing
- Create data versioning

Data labelling

- Data labeling can take a lot of time
- Use tools to make labeling easier
  - Vision
  - Sound
  - Text
  - Self made
- E.g., Label studio, Prodigy, LabelBox, time series annotator …

Prodi.gy

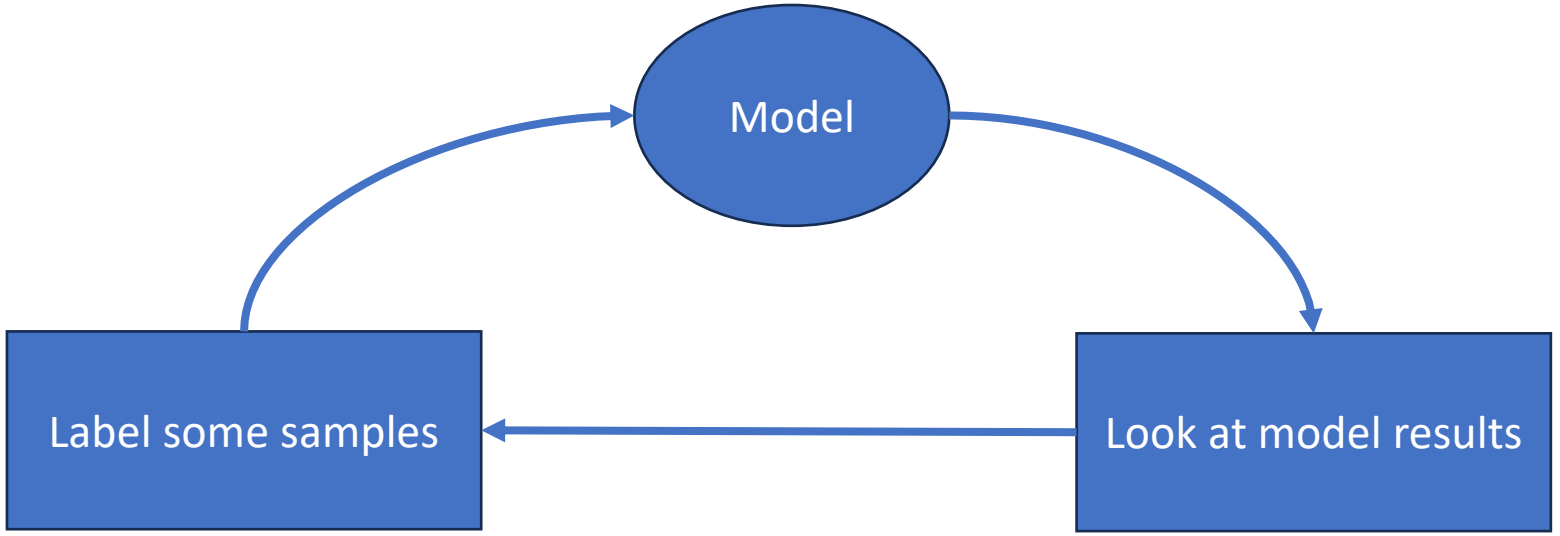
Time series annotator

Labelbox

Label Studio Community Edition

Use machine learning to help label data

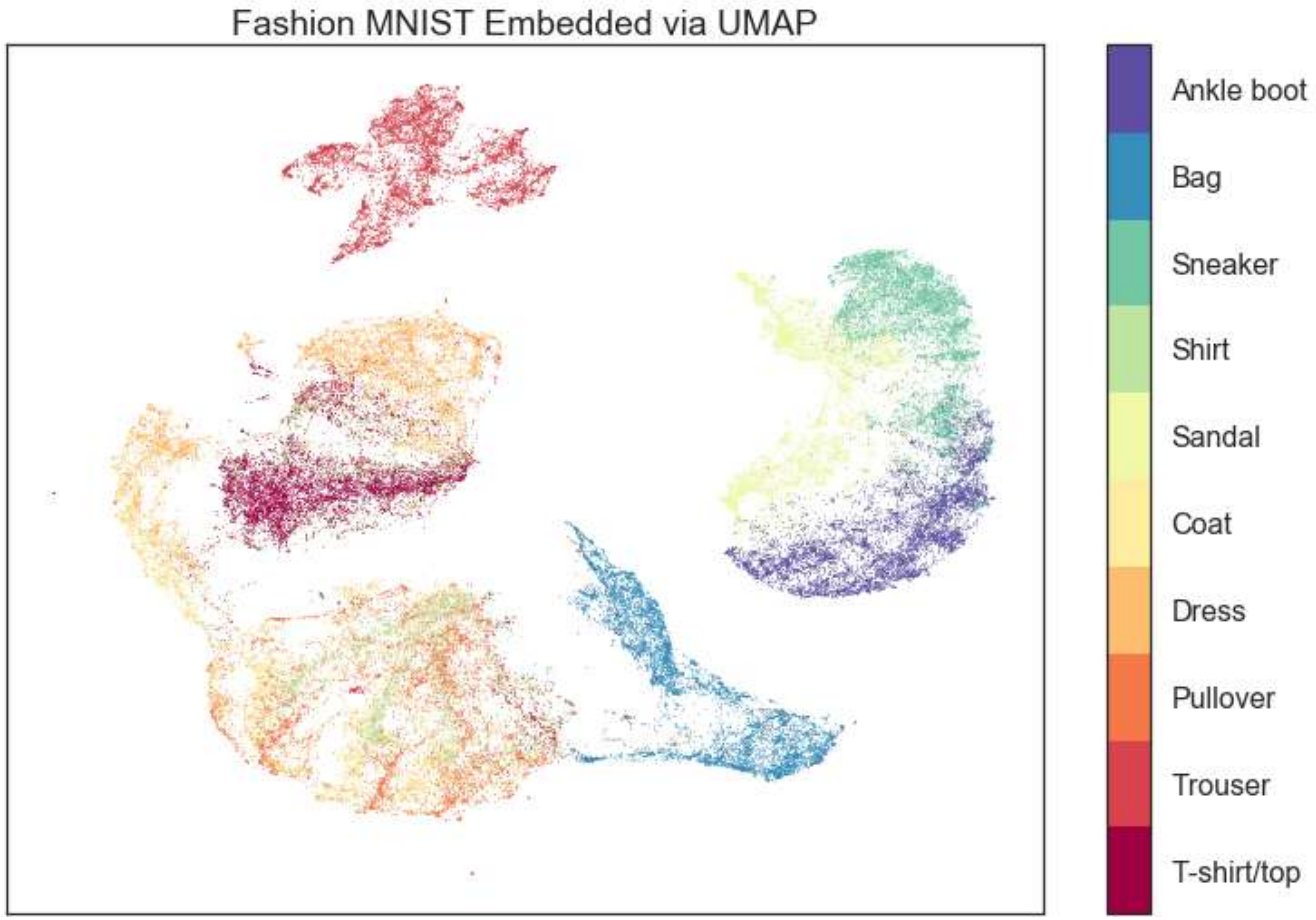
- Use a partially trained model to create labels
- Based on a trained model : which inputs would be best to label now?
  - Inputs of which the model is not sure
  - Inputs with biggest change if prediction is wrong
  - Inputs that are underrepresented
  - …



Dimensionality reduction can help find similar inputs

→ might have the same label

- UMAP
- t-SNE



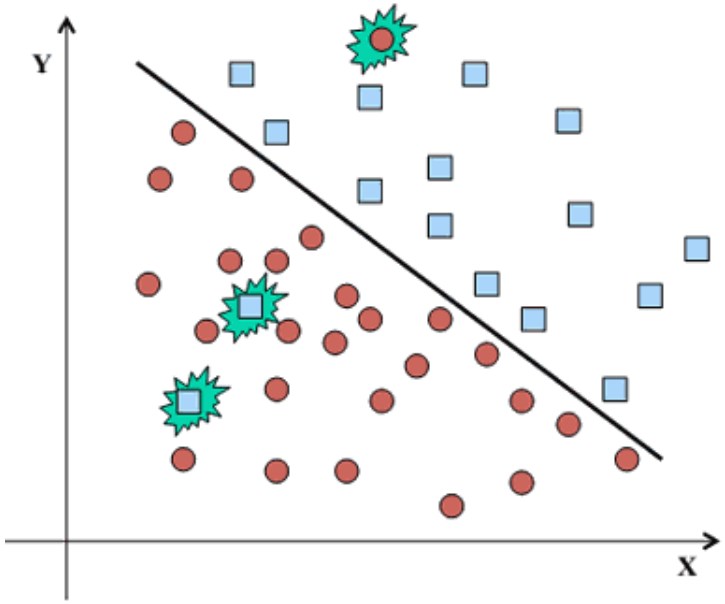
What?

- Collect the data
- Data labelling
- **Data preprocessing**
- Create data versioning

Data Cleaning

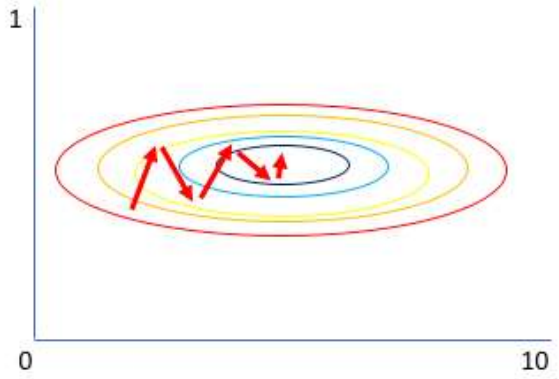
- Missing data: NaN, Null, … → remove, replace with mean …
- Noisy data → filtering, outlier removal, cluster analysis …

Column 0	age	years_seniority	income	parking_space	attending_party	entree	pets	emergency_contact
Tony	48	27		1	5	shrimp		Pepper
Donald	67	25	86	10	2	beef		Jane
Henry	69	21	95	6	1	chicken	62	Janet
Janet	62	21	110	3	1	beef		Henry
Nick		17		4				
Bruce	37	14	63		1	veggie		NA
Steve	83		77	7	1	chicken		n/a
Clint	27	9	118	9		shrimp	3	None
Wanda	19	7	52	2	2	shrimp		empty
Natasha	26	4	162	5	3			-
Carol		3	127	11	1	veggie	1	****
Mandy	44	2	68	8	1	chicken		null

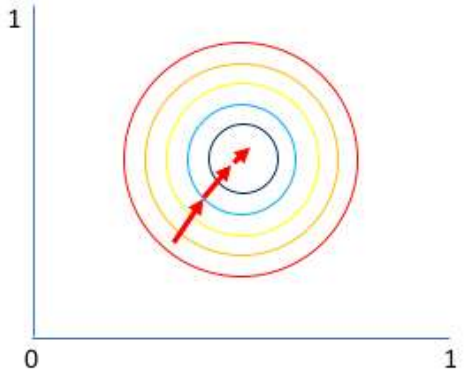


Data Transformation

- Normalization: min-max, z-score, linear …



Gradient of larger parameter dominates the update



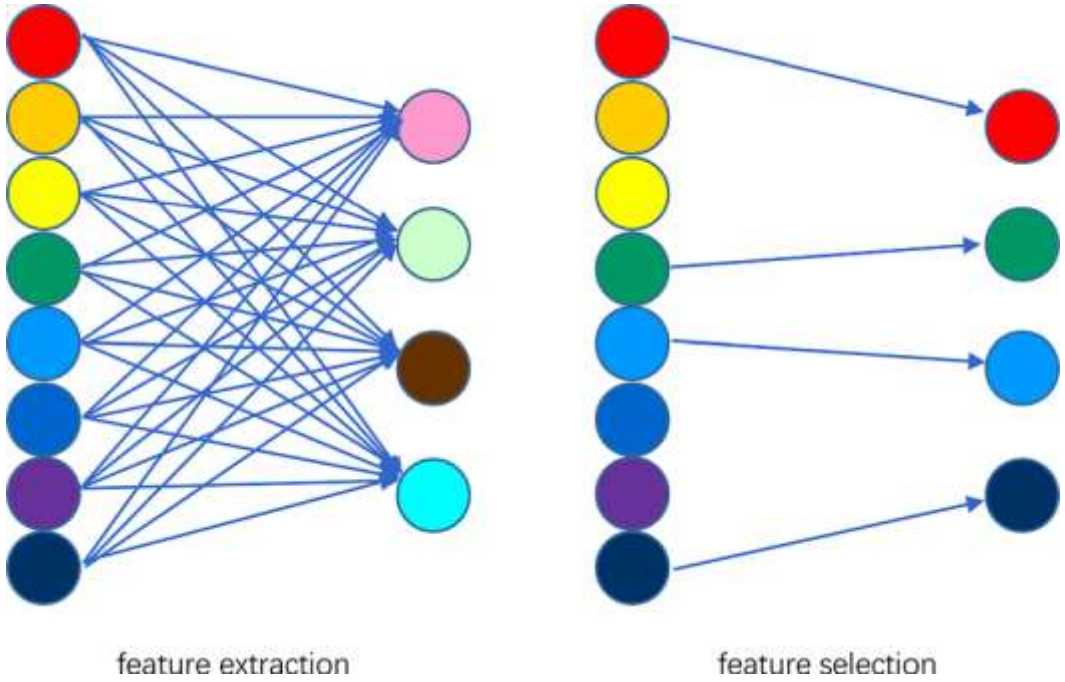
Both parameters can be updated in equal proportions

$$X^m = \frac{X - X_{min}}{X_{max} - x_{min}}$$

$$X^z = \frac{X - \mu}{\sigma}$$

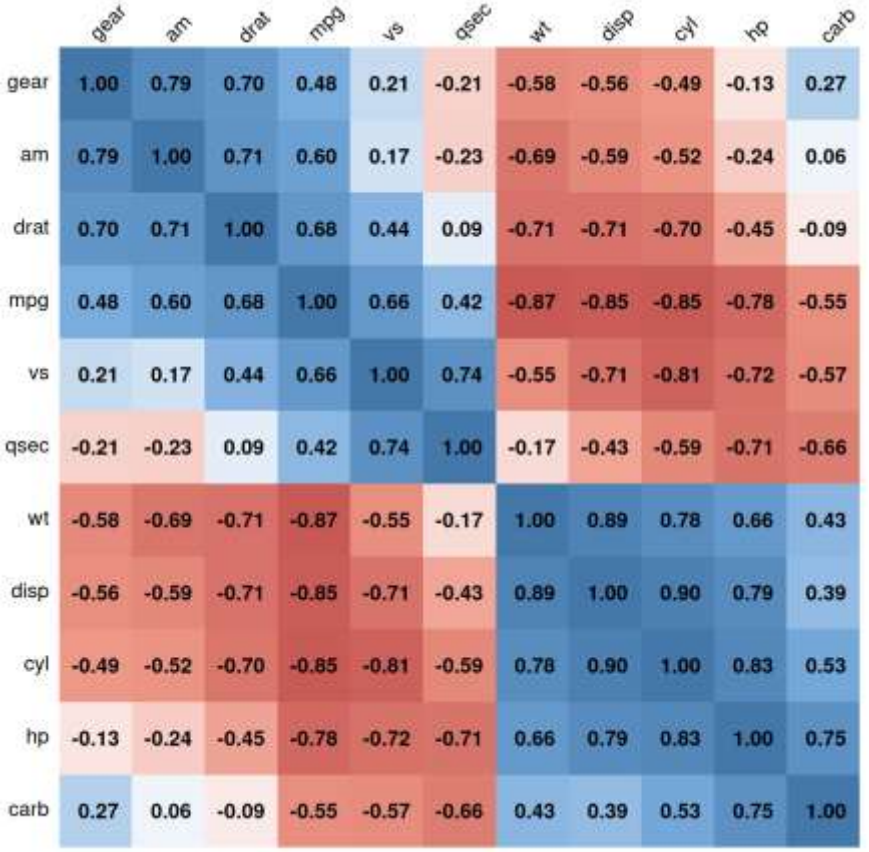
Data preprocessing - dimensionality reduction

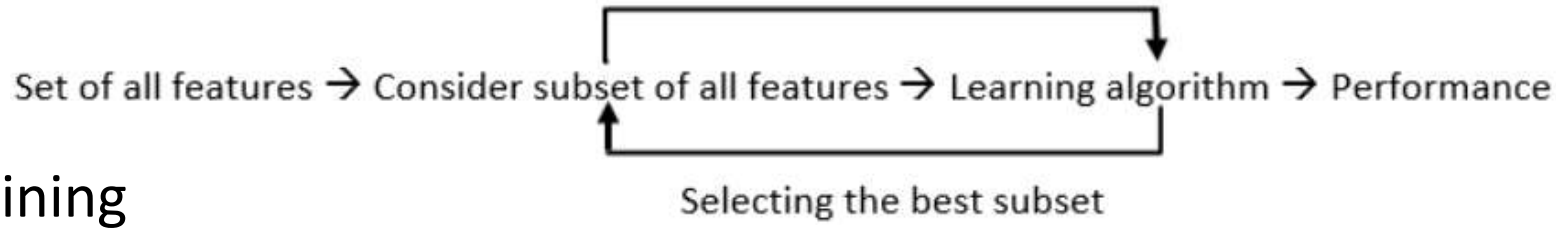
- Not all features are as important, or they hold similar information
- Feature selection
  - Select only the most relevant features
- Feature extraction
  - Create a singular new feature using the given features



Feature selection

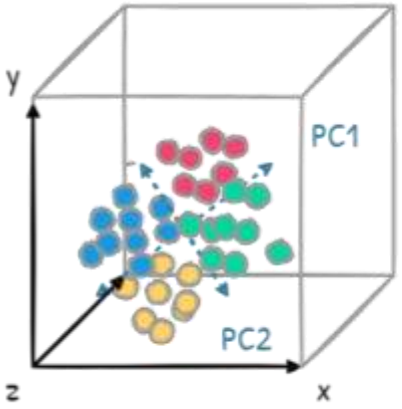
- Filter methods
  - Choose feature depending on metric
  - e.g. Correlation coefficient
- Wrapper methods
  - Select the best subset
  - Learn the algorithm
  - Select new subset
  - E.g. forward selection
- Embedded methods
  - Choose subset during training
  - L1 regularization

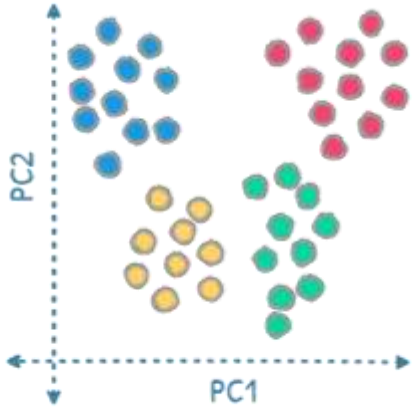




Feature extraction

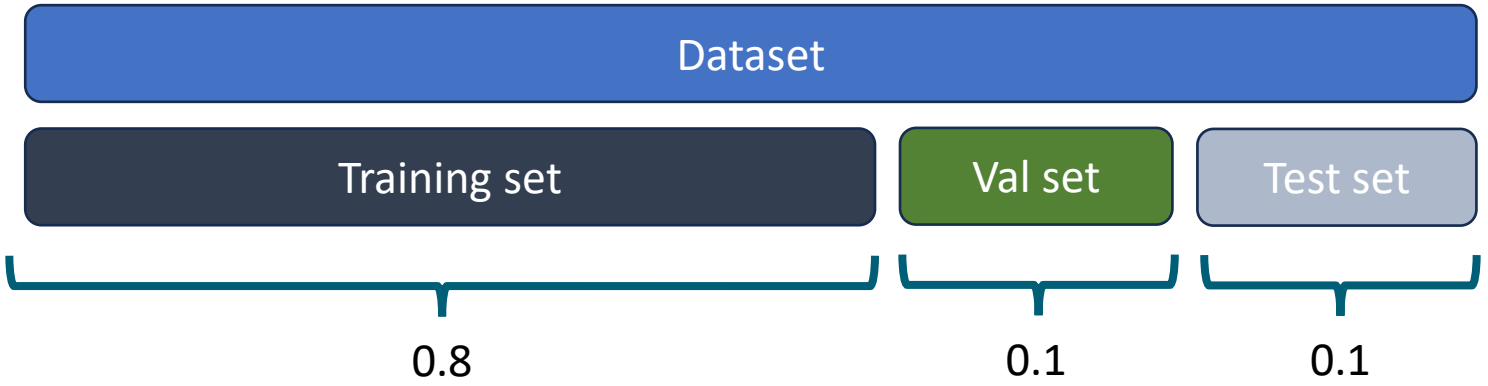
- Statistical methods:
  - mean, median, standard deviation, ...
- Dimensionality reduction:
  - Principle component analysis (PCA), Autoencoders, …
- Signal processing methods
  - Fourier transform, …
- Etc.





Data preprocessing

- Data augmentation
- Data splitting



Data preprocessing Tools

- Pandas:
  - Create and process Dataframes
- Scikit-learn:
  - Some specific data transformations
  - Split the data
- Pytorch
  - Some data transformations
  - Dataloaders to load data whilst training

Data visualisation

- Matplotlib: standard figure library
- Plotly: easy interactive plots
- Seaborn: easy Pandas dataframe graphs

Data dashboarding tools

- Sometimes you need a dashboard to have a good overview of your data
- Plotly Dash:
  - Open source
  - python library
  - creates HTML dashboards with plotly images
  - More customizable
- Streamlit:
  - Open source
  - Python library
  - App framework to build HTML pages
  - Easy and less code

What?

- Collect the data
- Data labelling
- Data preprocessing
- **Create data versioning**

Data tools

- Data labeling
- Data dashboards
- Data version control tools

Data versioning

- Code versioning tools like Git cannot deal with big files and data lakes
- Still need a way to keep track of the data used

Data Version Control

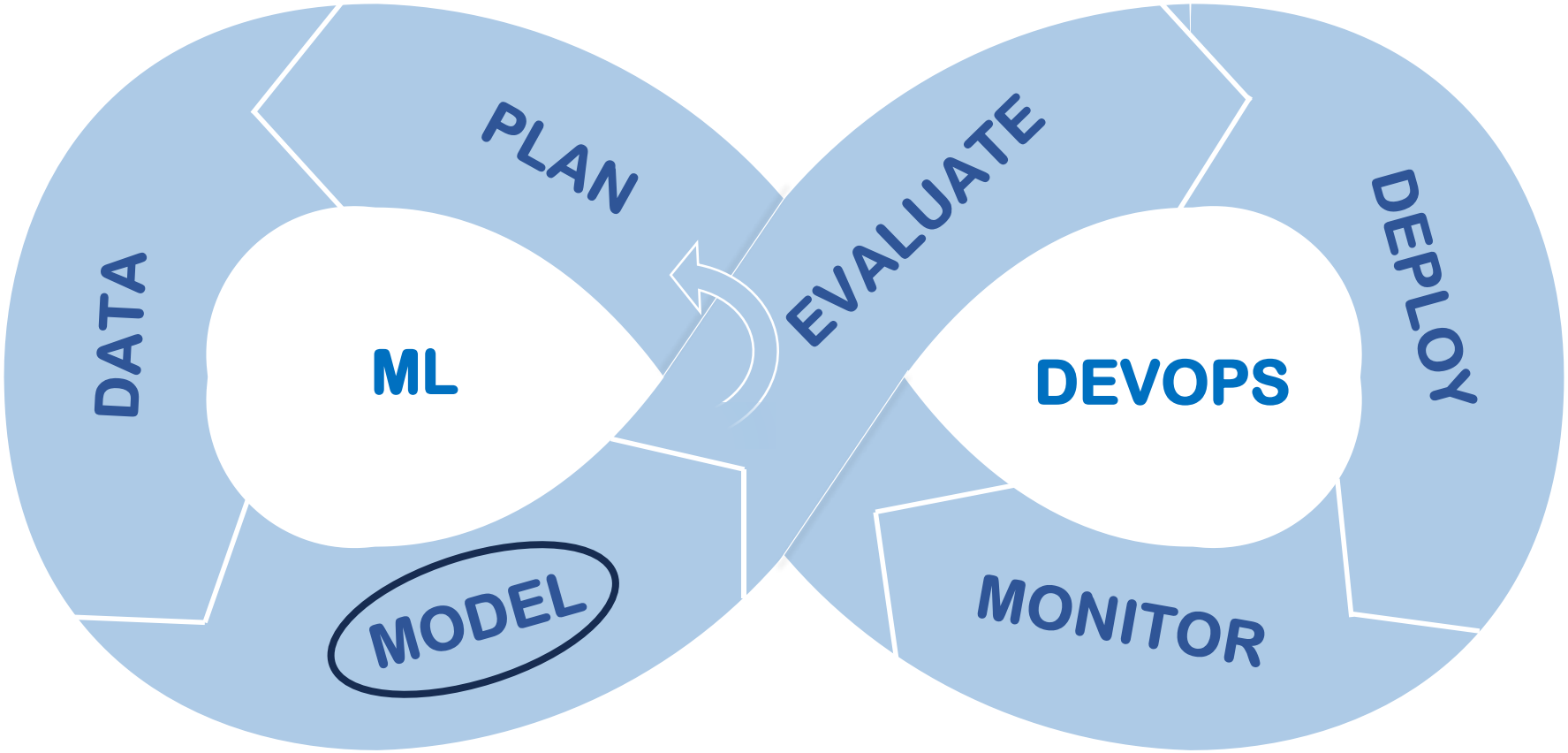
- Open-source data versioning tool (2017)
- Easy to use + lightweight
- Extension of Git with simple command line commands
- Has functionality to manage pipelines and ML models

VLAIO TETRA : MLOps4ECM
Model

Met steun van VLAIO

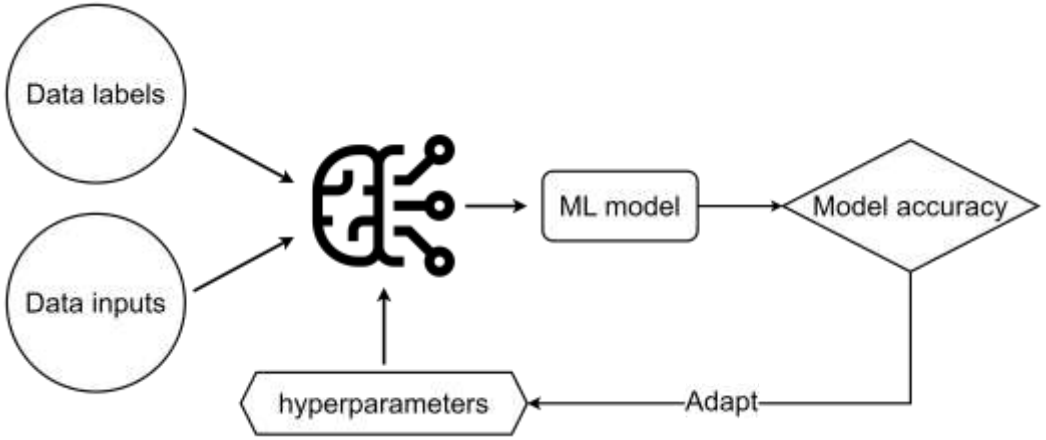
HBC.2023.0062

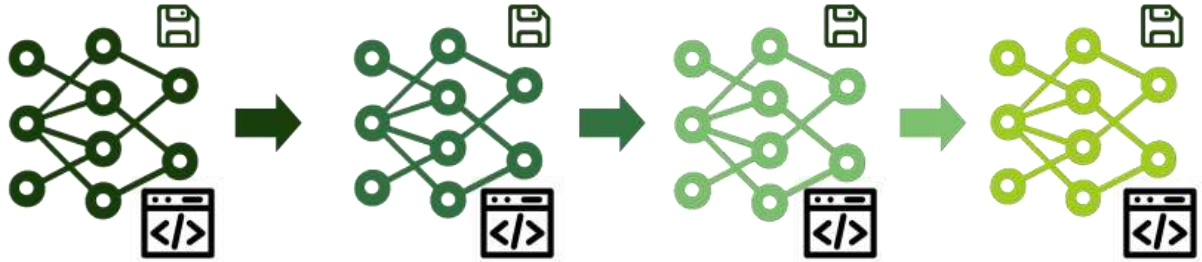
MLOps: pipeline



What

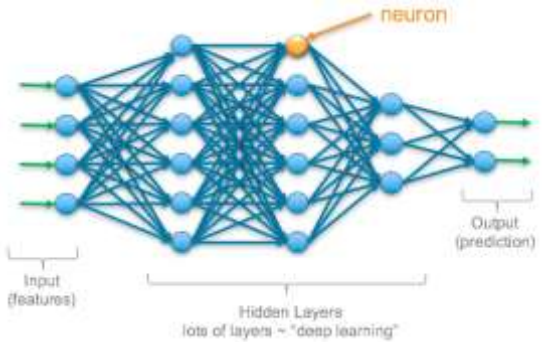
- Model architecture creation
- Model training
- Model finetuning
- Create model + metadata versioning

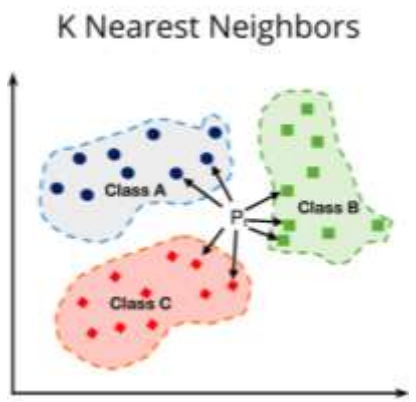


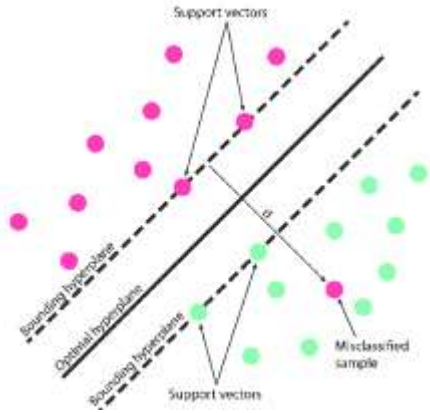


Architecture creation

- Which model:
  - (Deep) Neural Network
  - Support vector machine
  - K-nearest neighbours
  - …
- How does the model look
  - Number of hidden layers
  - size of the layers
  - Value of k
  - …

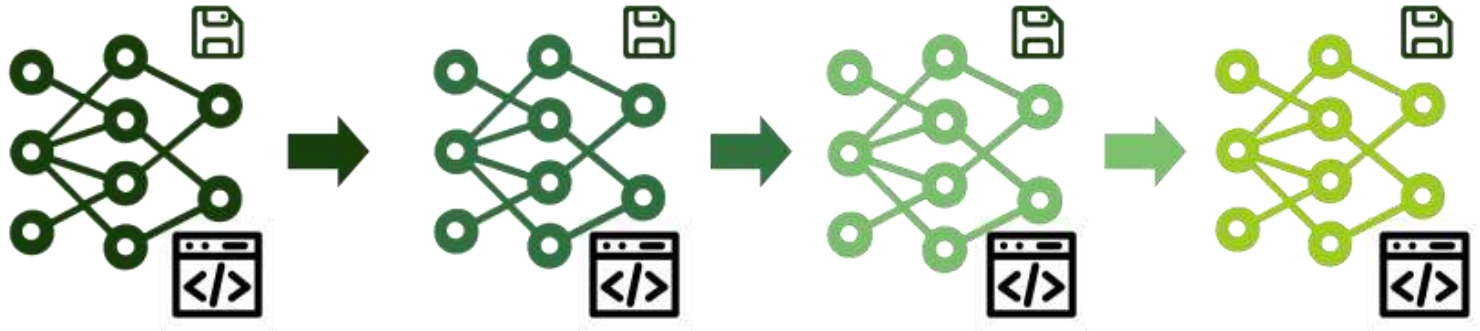






Hyperparameter tuning finds optimal combination of …

- Model parameters
  - See previous slide
- Training parameters
  - Learning rate
  - Loss function
  - Batch size
  - …

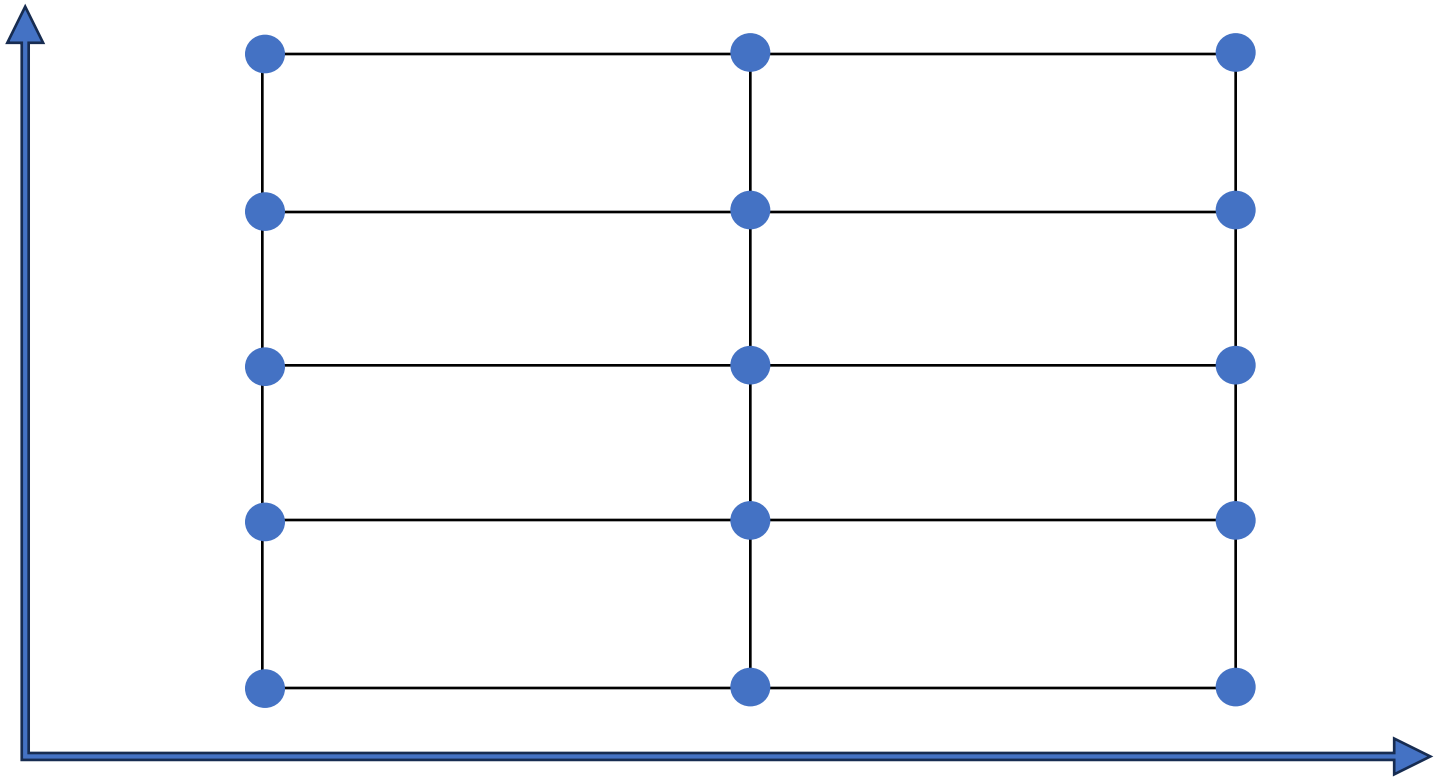


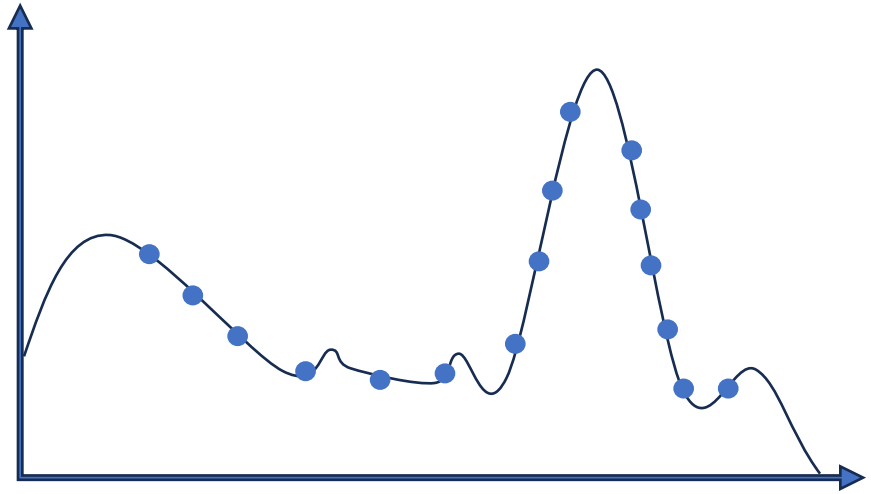
Hyperparameter tuning automation

- hyperparameter tuning can take a lot of time
- Needs to be done in a structured way

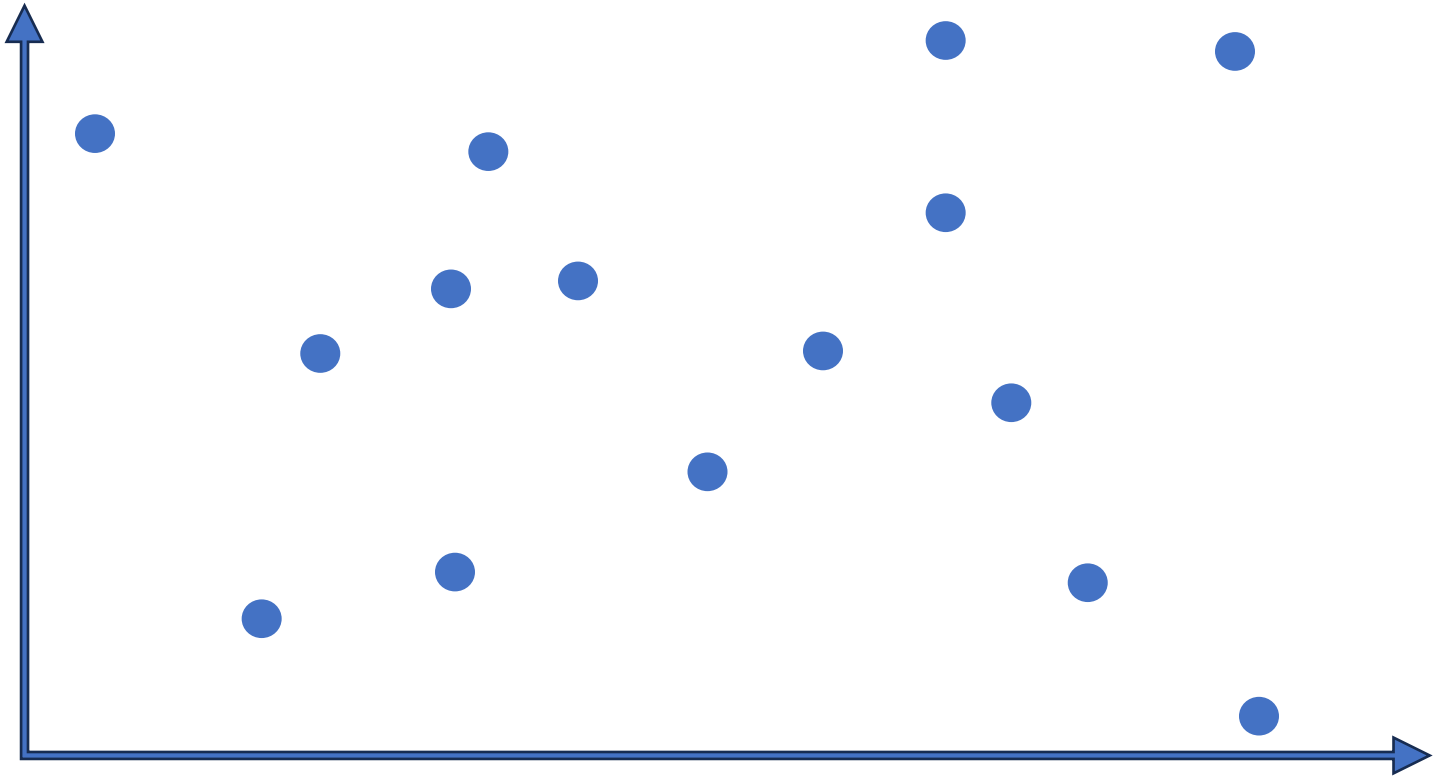
- Use of AutoML → automated search of hyperparameter tuning
  - Grid search
  - Random search
  - Bayesian optimization
  - Tree-structured Parzen Estimator

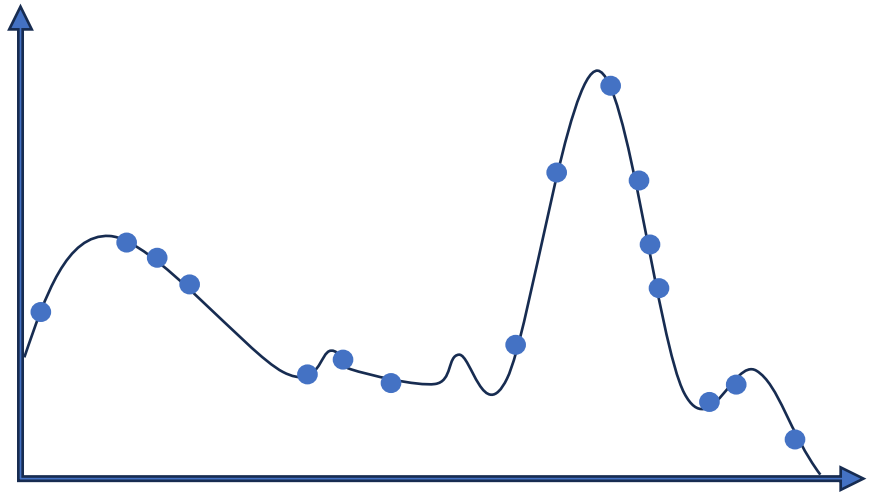
Grid search





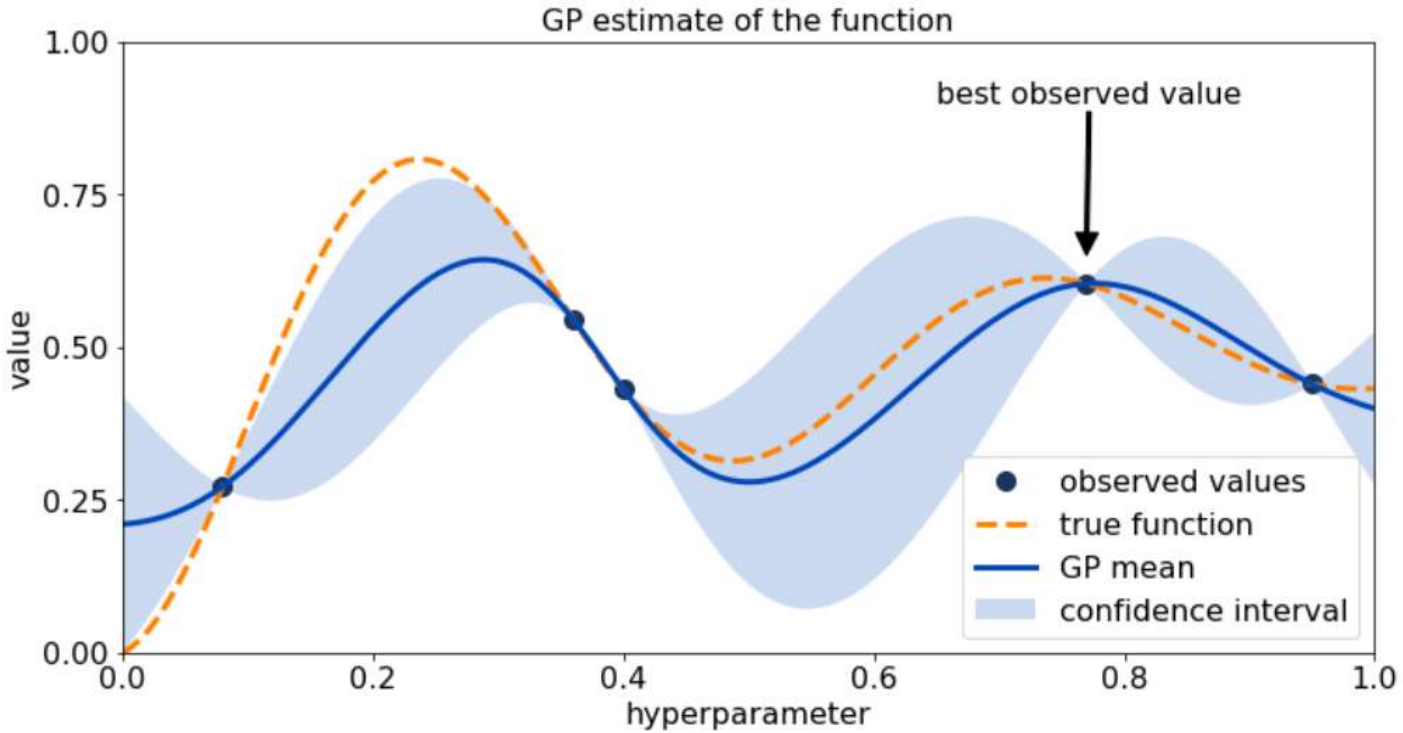
Random search





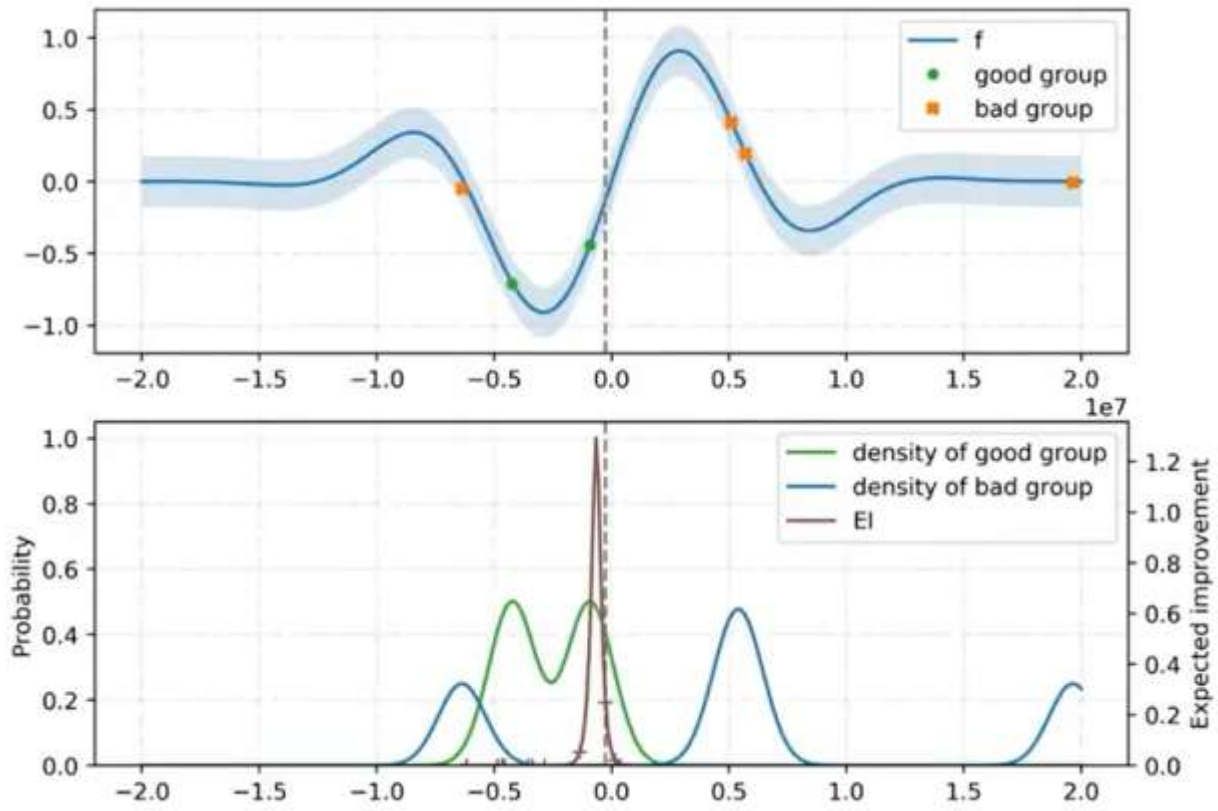
Bayesian Optimization

- Previous searches neglect previous observations



Tree-structured Parzen Estimator

- Two surrogate distribution
  - Bad values: loss worse than some threshold
  - Good values: loss better than some threshold
- Good hyperparameter:
  - low bad probability
  - high good probability



Optuna

- Open source
- Hyperparameter optimization framework
- Python library
- Choice in search and pruning algorithms
- Parallelization possible

Pruning – Median Pruner

Trial 1	Trial 2	Trial 3	Median	Trial 4
0.56	0.48	0.58	**0.56**	
0.55	0.48	0.56	**0.55**	
0.54	0.47	0.55	**0.54**	
0.52	0.46	0.53	**0.53**	
0.50	0.46	0.51	**0.50**	
0.49	0.45	0.47	**0.47**	
0.48	0.43	0.46	**0.46**	
0.48	0.42	0.43	**0.43**	
0.47	0.42	0.42	**0.42**	
0.46	0.41	0.41	**0.41**	

Pruning – Median Pruner

Trial 1	Trial 2	Trial 3	Median	Trial 4
0.56	0.48	0.58	**0.56**	0.55
0.55	0.48	0.56	**0.55**	
0.54	0.47	0.55	**0.54**	
0.52	0.46	0.53	**0.53**	
0.50	0.46	0.51	**0.50**	
0.49	0.45	0.47	**0.47**	
0.48	0.43	0.46	**0.46**	
0.48	0.42	0.43	**0.43**	
0.47	0.42	0.42	**0.42**	
0.46	0.41	0.41	**0.41**	

Pruning – Median Pruner

Trial 1	Trial 2	Trial 3	Median	Trial 4
0.56	0.48	0.58	**0.56**	0.55
0.55	0.48	0.56	**0.55**	0.54
0.54	0.47	0.55	**0.54**	
0.52	0.46	0.53	**0.53**	
0.50	0.46	0.51	**0.50**	
0.49	0.45	0.47	**0.47**	
0.48	0.43	0.46	**0.46**	
0.48	0.42	0.43	**0.43**	
0.47	0.42	0.42	**0.42**	
0.46	0.41	0.41	**0.41**	

Pruning – Median Pruner

Trial 1	Trial 2	Trial 3	Median	Trial 4
0.56	0.48	0.58	**0.56**	0.55
0.55	0.48	0.56	**0.55**	0.54
0.54	0.47	0.55	**0.54**	0.53
0.52	0.46	0.53	**0.53**	
0.50	0.46	0.51	**0.50**	
0.49	0.45	0.47	**0.47**	
0.48	0.43	0.46	**0.46**	
0.48	0.42	0.43	**0.43**	
0.47	0.42	0.42	**0.42**	
0.46	0.41	0.41	**0.41**	

Pruning – Median Pruner

Trial 1	Trial 2	Trial 3	Median	Trial 4
0.56	0.48	0.58	**0.56**	0.55
0.55	0.48	0.56	**0.55**	0.54
0.54	0.47	0.55	**0.54**	0.53
0.52	0.46	0.53	**0.53**	0.52
0.50	0.46	0.51	**0.50**	
0.49	0.45	0.47	**0.47**	
0.48	0.43	0.46	**0.46**	
0.48	0.42	0.43	**0.43**	
0.47	0.42	0.42	**0.42**	
0.46	0.41	0.41	**0.41**	

Pruning – Median Pruner

Trial 1	Trial 2	Trial 3	Median	Trial 4
0.56	0.48	0.58	**0.56**	0.55
0.55	0.48	0.56	**0.55**	0.54
0.54	0.47	0.55	**0.54**	0.53
0.52	0.46	0.53	**0.53**	0.52
0.50	0.46	0.51	**0.50**	0.51
0.49	0.45	0.47	**0.47**	
0.48	0.43	0.46	**0.46**	
0.48	0.42	0.43	**0.43**	
0.47	0.42	0.42	**0.42**	
0.46	0.41	0.41	**0.41**	

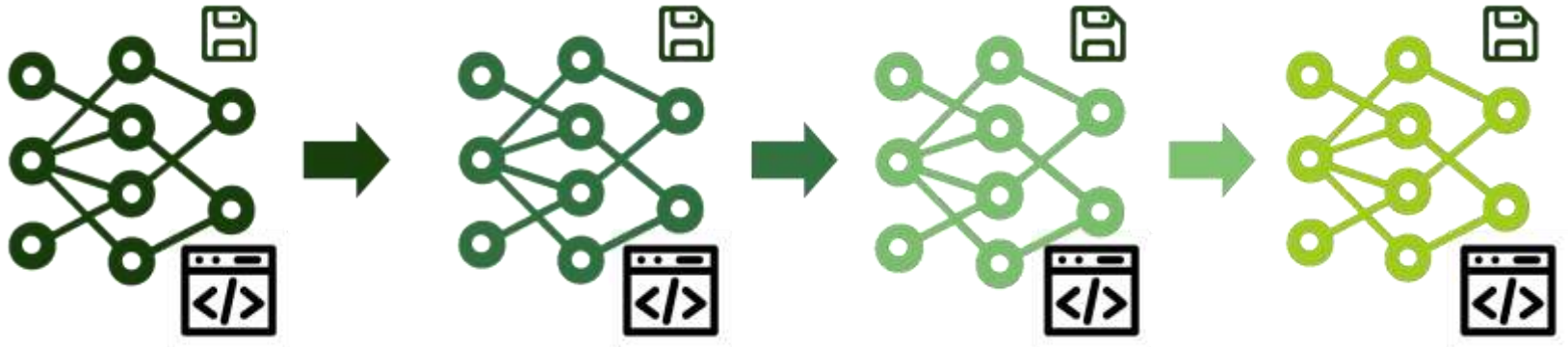
Pruning – Median Pruner

Trial 1	Trial 2	Trial 3	Median	Trial 4
0.56	0.48	0.58	**0.56**	0.55
0.55	0.48	0.56	**0.55**	0.54
0.54	0.47	0.55	**0.54**	0.53
0.52	0.46	0.53	**0.53**	0.52
0.50	0.46	0.51	**0.50**	0.51
0.49	0.45	0.47	**0.47**	
0.48	0.43	0.46	**0.46**	Prune!
0.48	0.42	0.43	**0.43**	
0.47	0.42	0.42	**0.42**	
0.46	0.41	0.41	**0.41**	

Whilst finetuning → different model versions

- Hyperparameters (learning rate, batch size …)
- Model architectures
- Data
- Metric results (accuracy, loss …)

  = METADATA



MLflow

- Open-source
- Manage ML lifecycle
- Python package
- MLflow model tracking (logging parameters)
- Mlflow projects organize implementation code in reproducible way
- Mlflow model packaging for serving
- Model registry : model versions and lineage

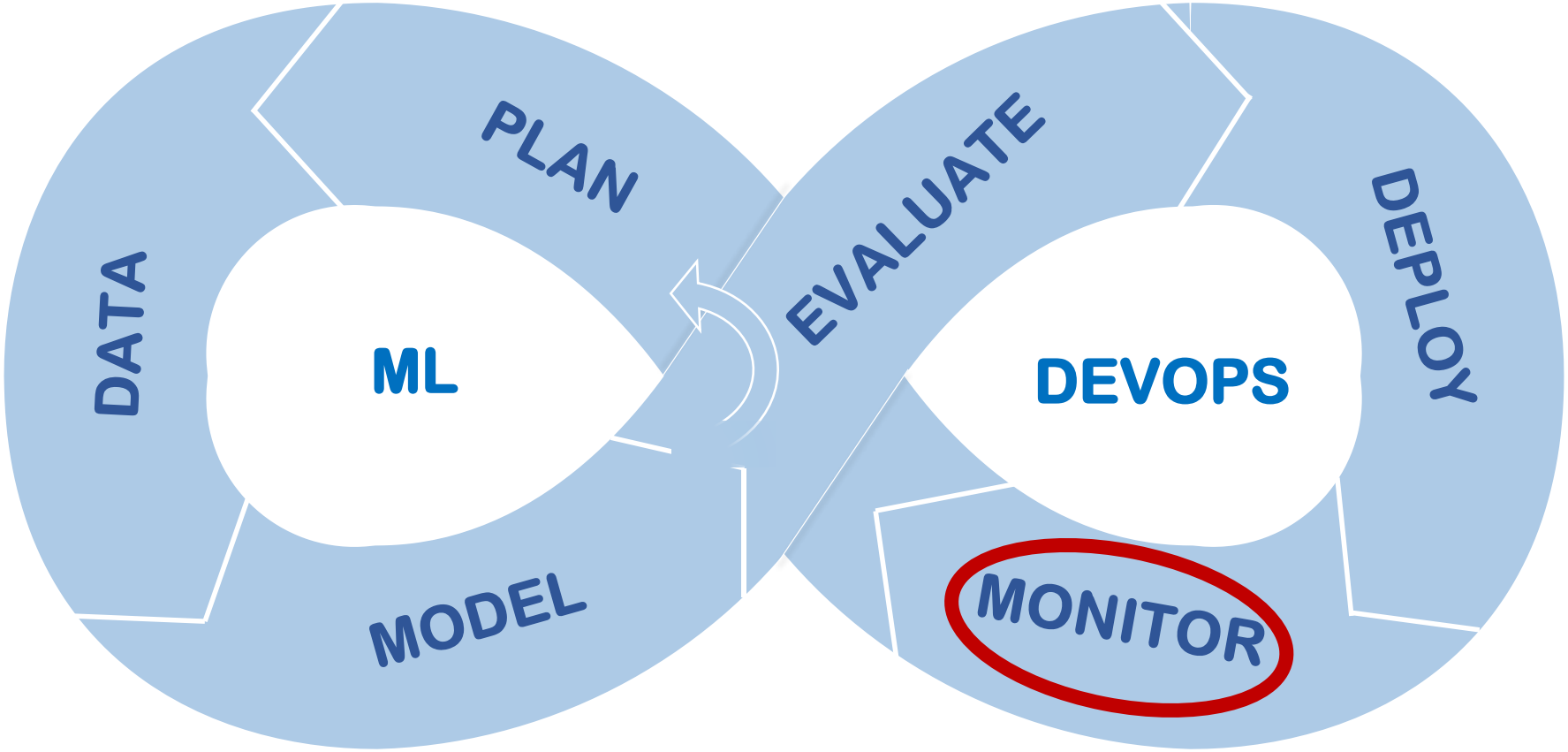
VLAIO TETRA : MLOps4ECM
Monitoring

Lara Luys

Met steun van VLAIO

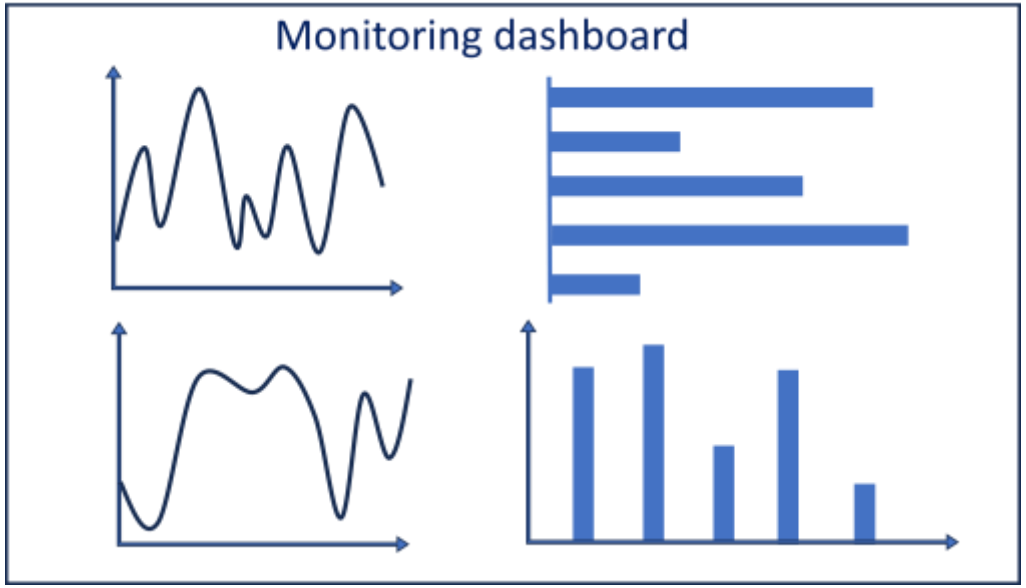
HBC.2023.0062

MLOps: pipeline



what is monitoring?

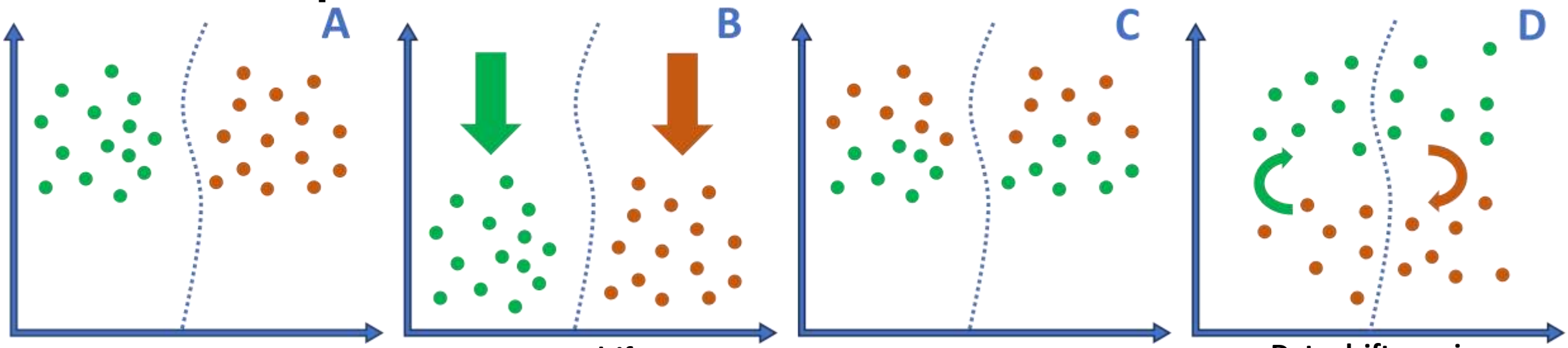
- Monitor the data for data quality and data drift
- Monitor the machine learning model for concept drift
- Monitor other parameters for drift e.g. resources, fairness …



Monitoring

- **What is drift**
- Monitoring tools
- Data quality
- Model quality
- Statistical tests
- Using machine learning models
- Other types of drift

Concept vs Data drift



Original data

Data drift (covariate drift)

Data distribution changes but the model prediction is still correct

e.g. classification dog vs. Cat gets cats in different colors

Concept drift

Data distribution stays the same, but the model prediction is incorrect

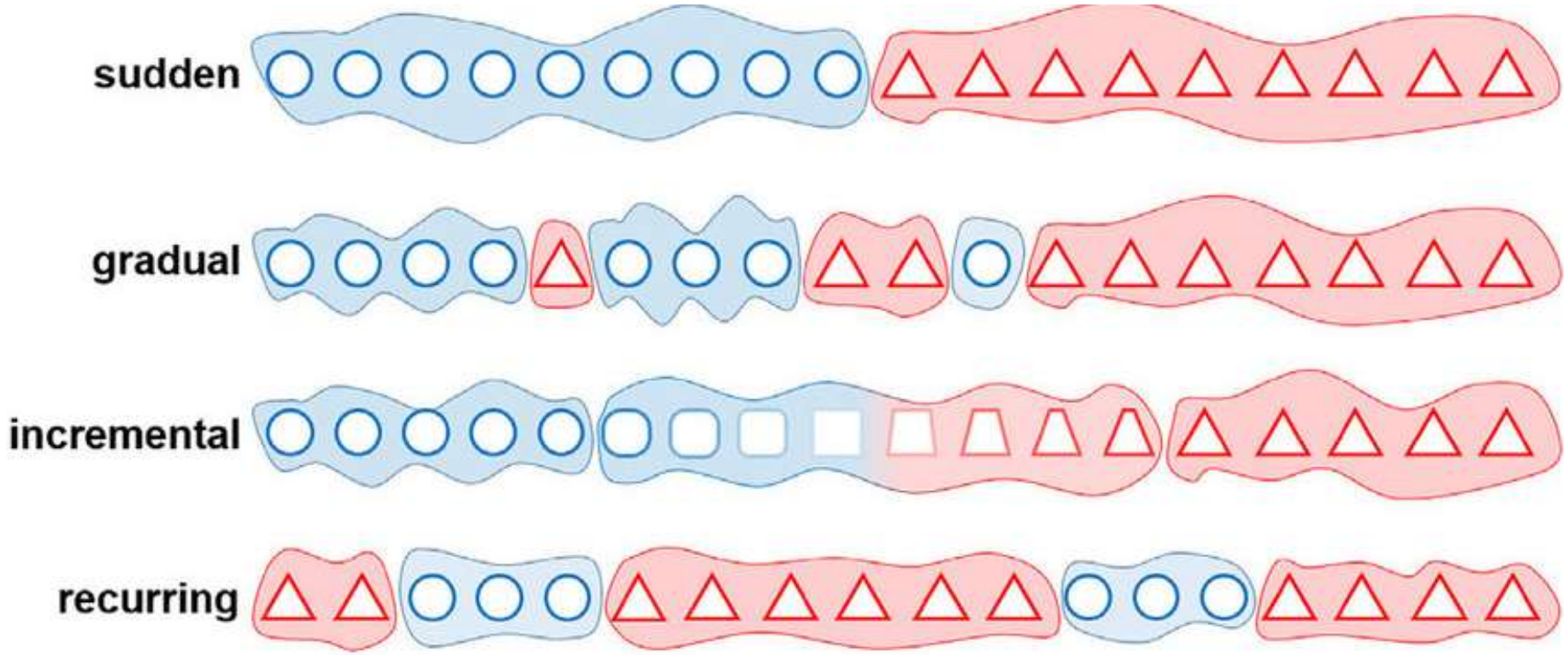
e.g. prediction housing price gets more expensive over time

Data drift causing concept drift

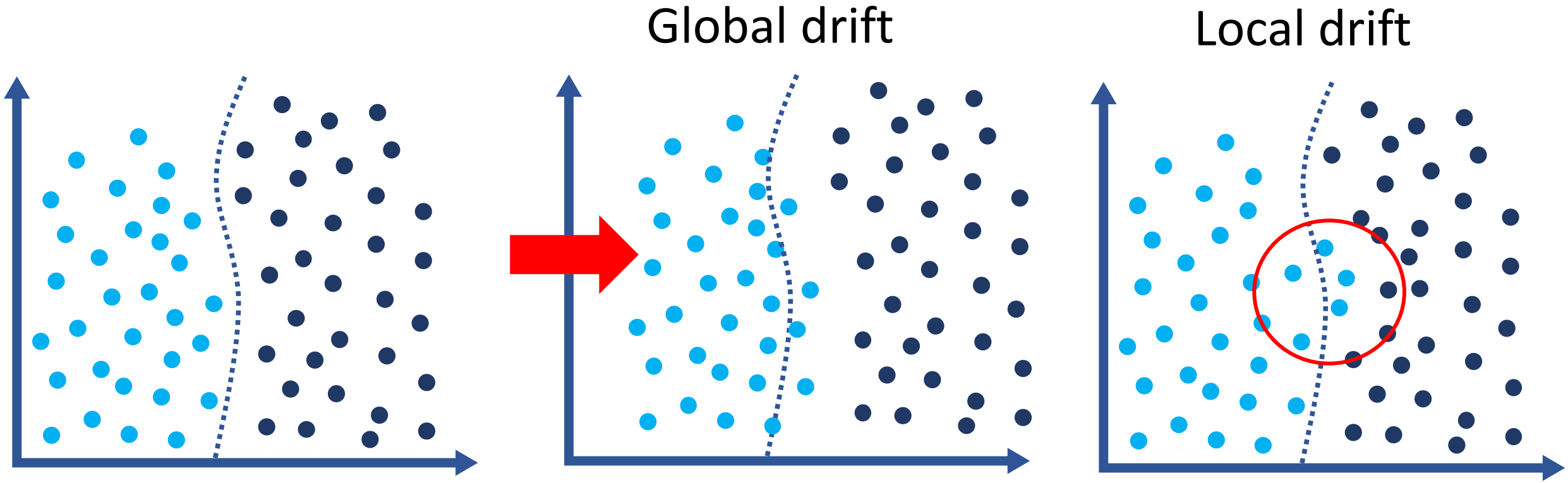
Data distribution changes and the model prediction is incorrect

e.g. different machine settings lead to wrong predictions

Types of drift : timing

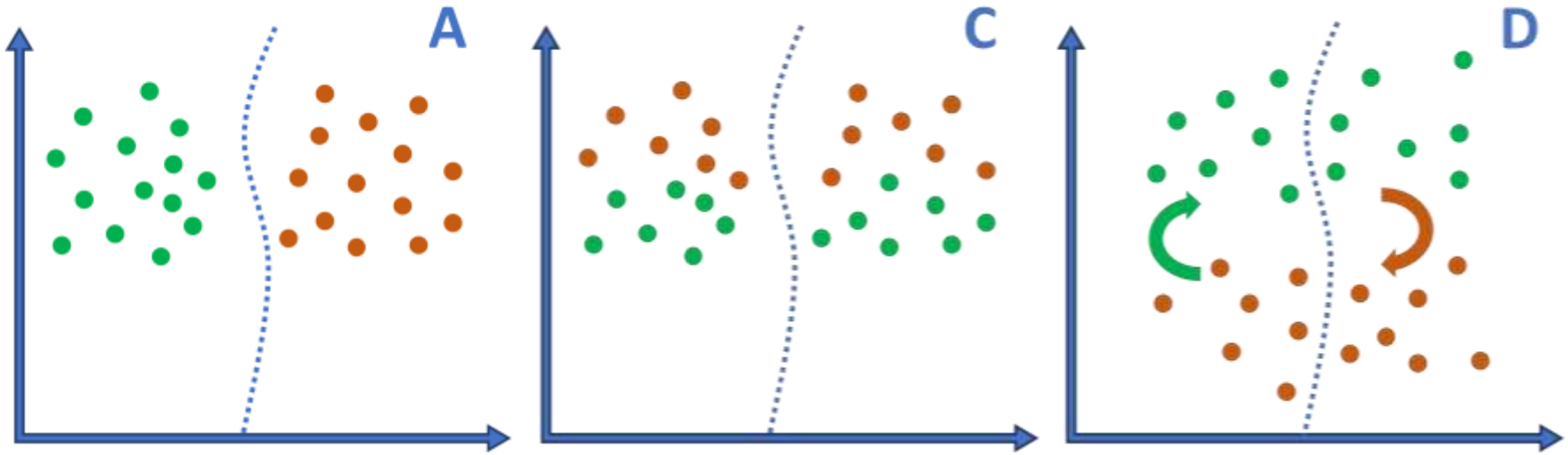


Types of drift: Location



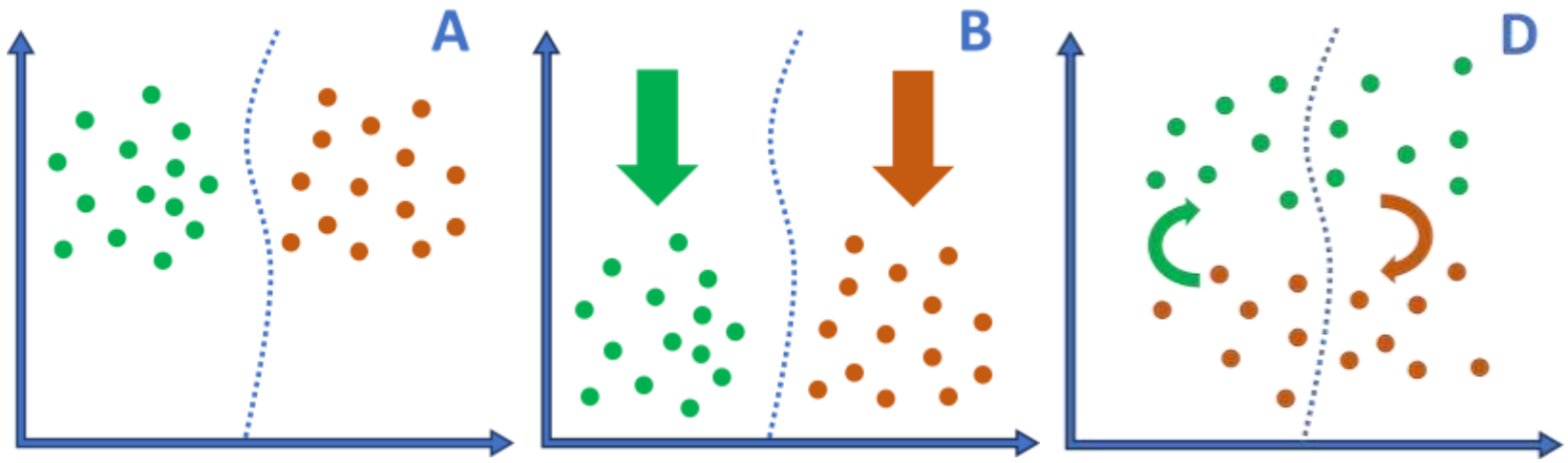
Concept drift detection

- The distribution of the output target changes
- If true **labels** available → monitor **model quality**
- Otherwise → **prediction** distribution monitoring



Data drift detection

- Input data distribution monitoring



- If you must choose → monitor predictions over input data

Monitoring

- What is drift
- **Monitoring tools**
- Data quality
- Model quality
- Statistical tests
- Using machine learning models
- Other types of drift

Types of tools that can help

- Dashboarding and logging tools
  - Prometheus, grafana
- Tools that monitor input data, predictions and model quality
  - Evidently, Arize, Whylabs ...

Grafana

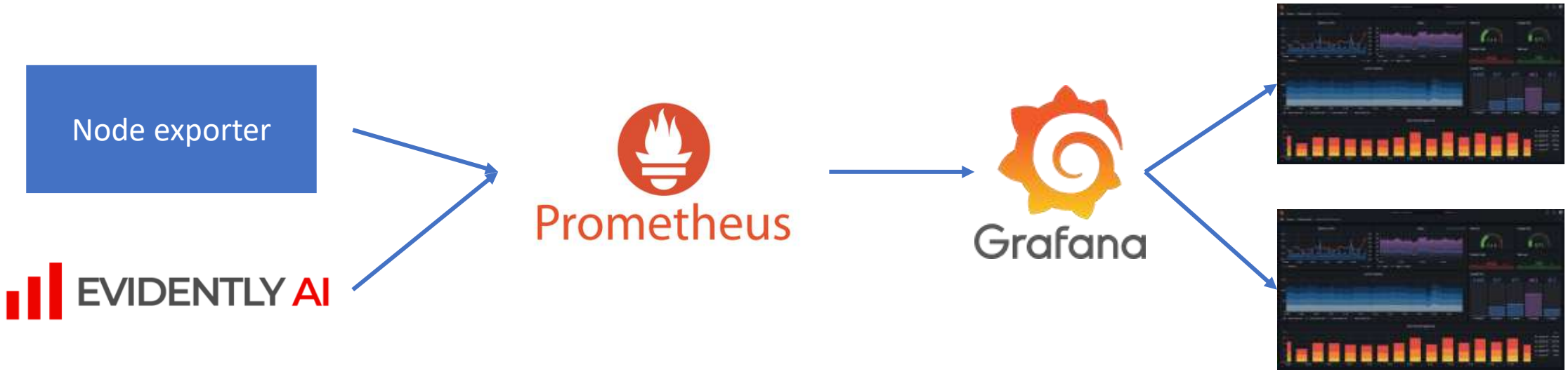
- Observability platform
- Create easy dashboards without code
- Cloud option available
- Lot of different data sources connections available

Prometheus

- Open-source system monitoring + alerting toolkit
- Collects time series data, logs and metrics
- Uses promQL as query language to query data
- Some graph and dashboarding features

Grafana + Prometheus: create monitoring dashboards

- Machine information (CPU, memory …)
- Application usage (clicks on certain pages)
- Machine learning model (metrics of model)
- …



Evidently

- Open-source ML observability platform
- Python library → create (pre-made) test suits and reports
- Able to save as html, json, dictionary → fully offline possible
- Mainly tabular and text data → working on other unstructured data

Monitoring

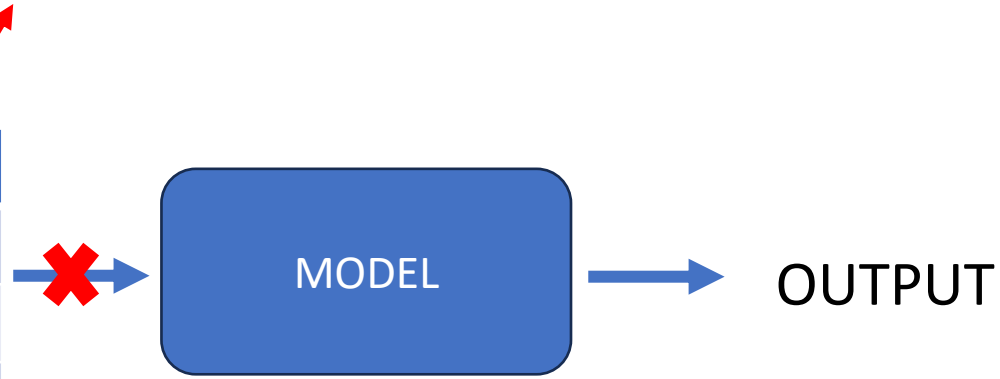
- What is drift
- Monitoring tools
- **Data quality**
- Model quality
- Statistical tests
- Using machine learning models
- Other types of drift

Monitoring - Data quality

- Use reference dataset (Training dataset)
  - Expected schema and column types
  - Expected batch size
  - Statistics: averages, min-max ...
  - …

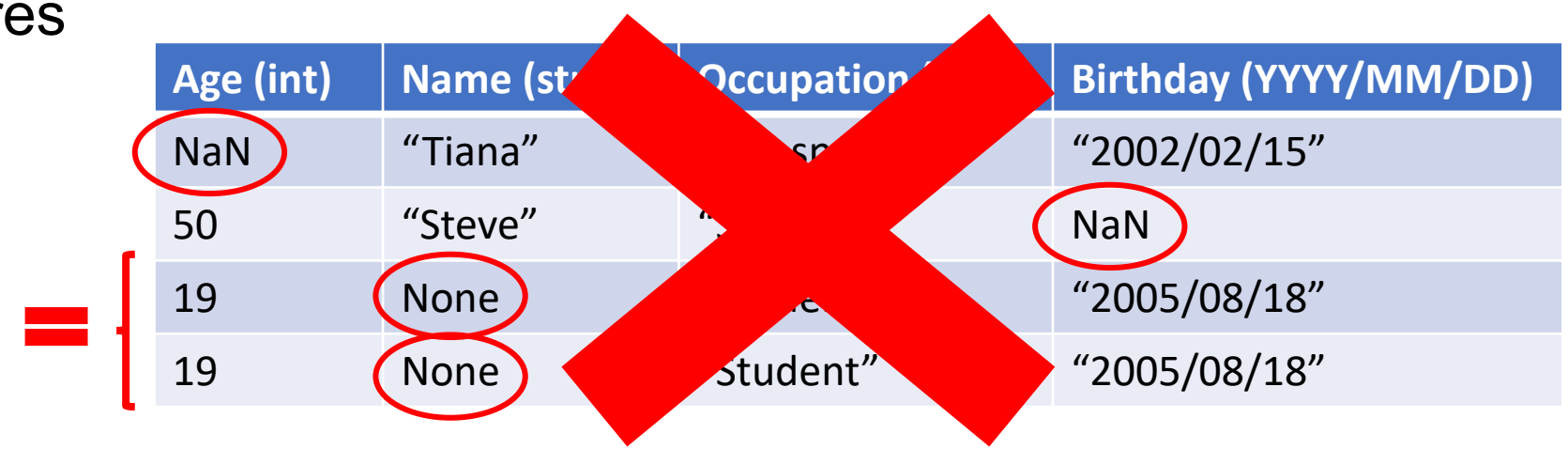
Age (int)	Name (str)	Occupation (str)	Birthday (DD/MM/YYYY)

Age (int)	Name (str)	Occupation (str)	Birthday (YYYY/MM/DD)
23	“Tiana”	“Salesperson”	“2002/02/15”
50	“Steve”	“Sailor”	“1975/01/07”
19	“Wanda”	“Student”	“2005/08/18”



Monitoring - Data quality

- Use reference dataset
- Use thresholds
  - Share of missing values
  - Duplicate columns/rows
  - Constant features
  - ...

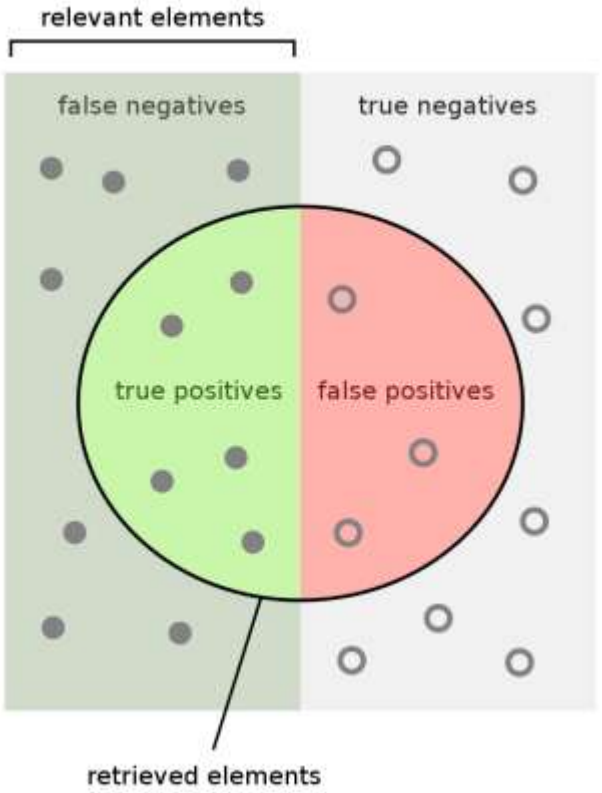


Monitoring

- What is drift
- Monitoring tools
- Data quality
- **Model quality**
- Statistical tests
- Using machine learning models
- Other types of drift

Monitoring - Model quality metrics

- Accuracy
- Recall
- Precision

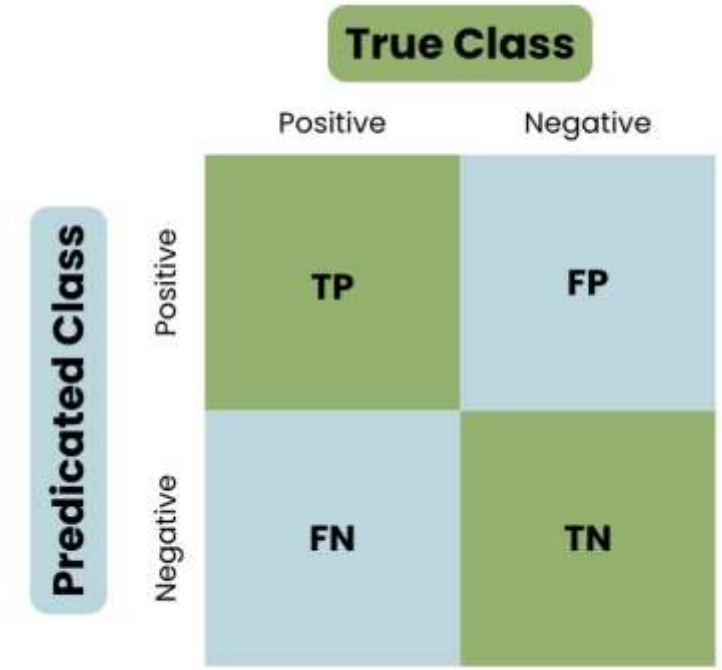


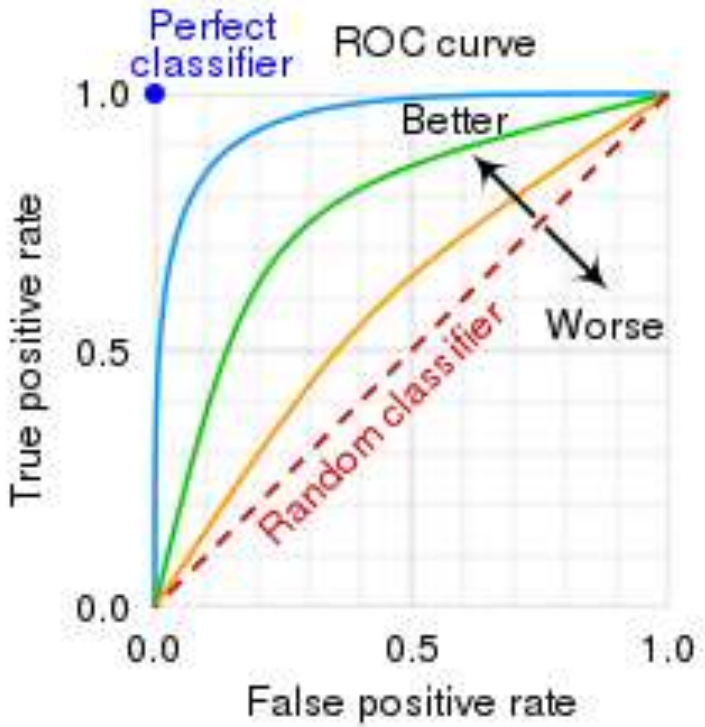
$$\text{Precision} = \frac{\text{green}}{\text{green + red}} \qquad \text{Recall} = \frac{\text{green}}{\text{green + dark green}}$$

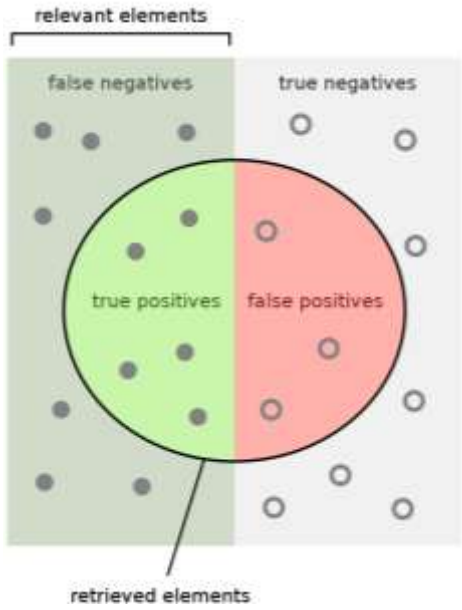
$$\text{Accuracy} = \frac{\text{green + white}}{\text{green + white + dark green + red}}$$

Monitoring - Model quality metrics

- Confusion matrix
- ROC

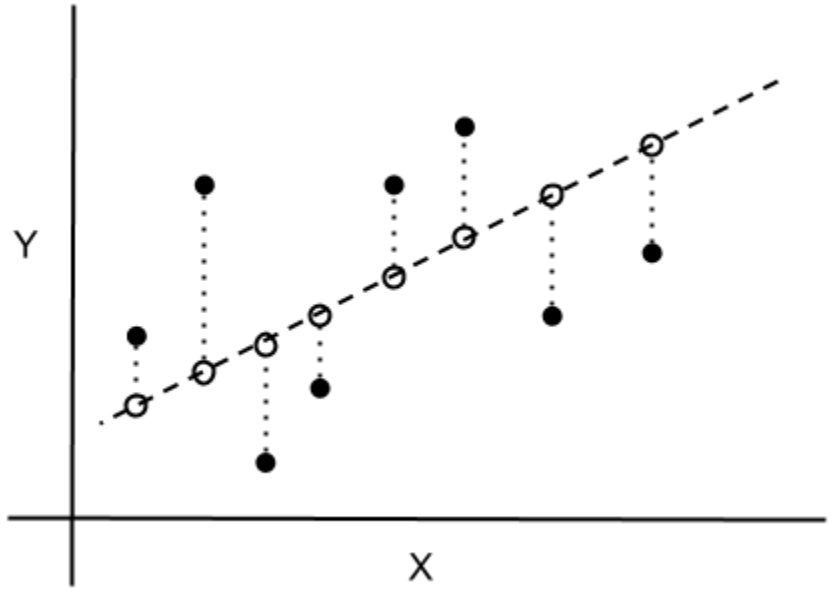






Monitoring - Model quality metrics

- Mean Squared Error (MSE)
- Mean Absolute Error (MAE)
- …

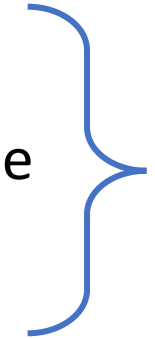


$$MSE = \frac{1}{n}\sum_{i=1}^{n}\left(Y_i - \hat{Y}_i\right)^2$$

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - \hat{Y}_i\right|$$

Monitoring - Model quality metrics

- Significant changes = Drift in the model
- E.g.
  - Accuracy = lower
  - Confusion matrix = worse
  - MSE = higher



Model perfoms worse

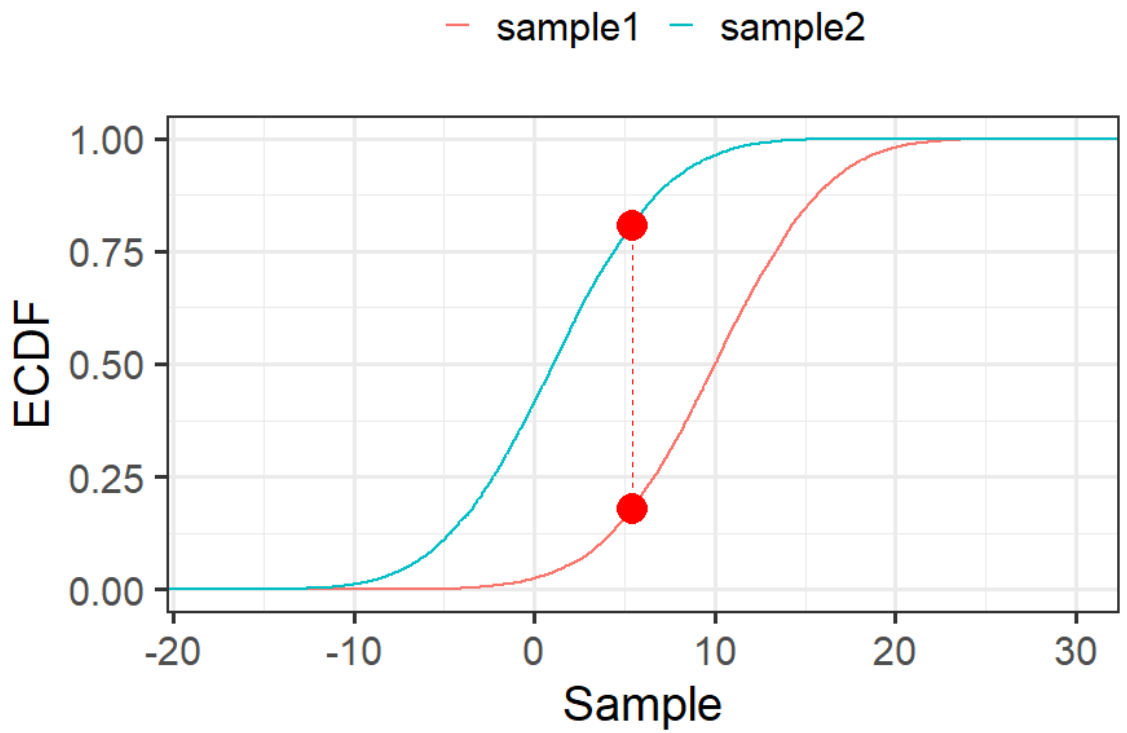
- Use treshold or reference dataset to compare

Monitoring

- What is drift
- Monitoring tools
- Data quality
- Model quality
- **Statistical tests**
- Using machine learning models
- Other types of drift

Statistical tests : Compare two statistical distributions (reference and current)

- Kolmogorov-Smirnov
  - Max distance of the cumulative distributions
  - Prone to false positives
  - Higher value is more differrent



Statistical tests : Compare two statistical distributions (reference and current)

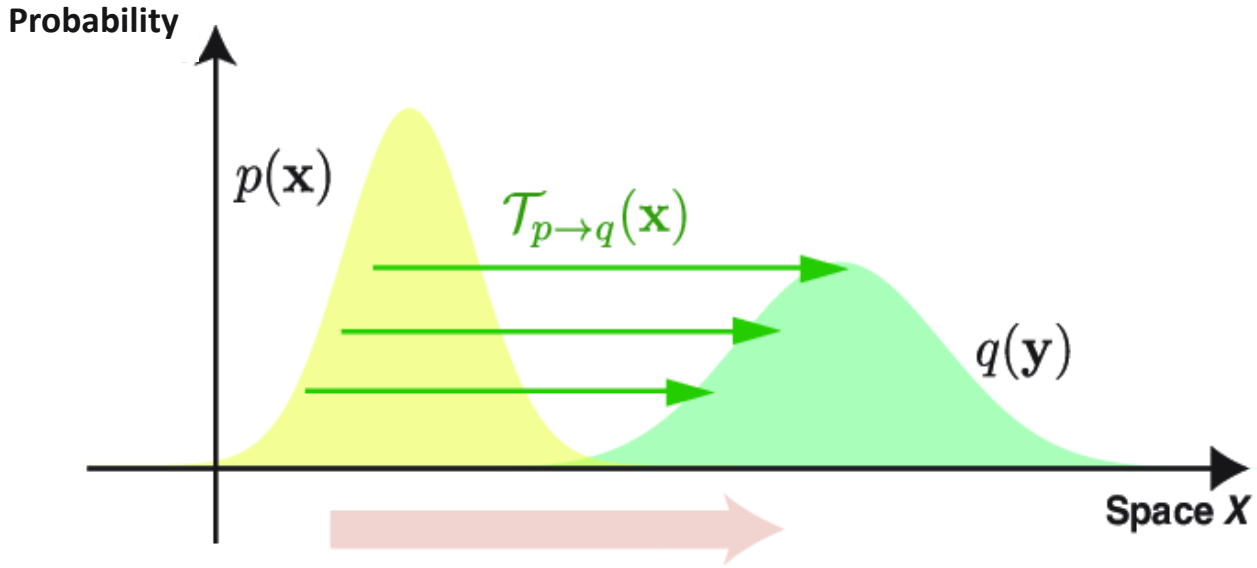
- Chi-squared
  - Categorical variables
  - Compare **difference** between **current** values and **expected** values for each category
  - Higher value = more different

$$X^2 = \sum \frac{(\text{Observed value} - \text{Expected value})^2}{\text{Expected value}}$$

	Phenotypes of offspring			
	brown coat, long ears	brown coat, short ears	black coat, long ears	Black coat, short ears
Observed number (O)	73	21	26	8
Expected ratio	9 : 3 : 3 : 1			
Expected number (E)	72	24	24	8
O – E	1	–3	2	0
$(O - E)^2$	1	9	4	0
$(O - E)^2 / E$	1/72	9/24	4/24	0

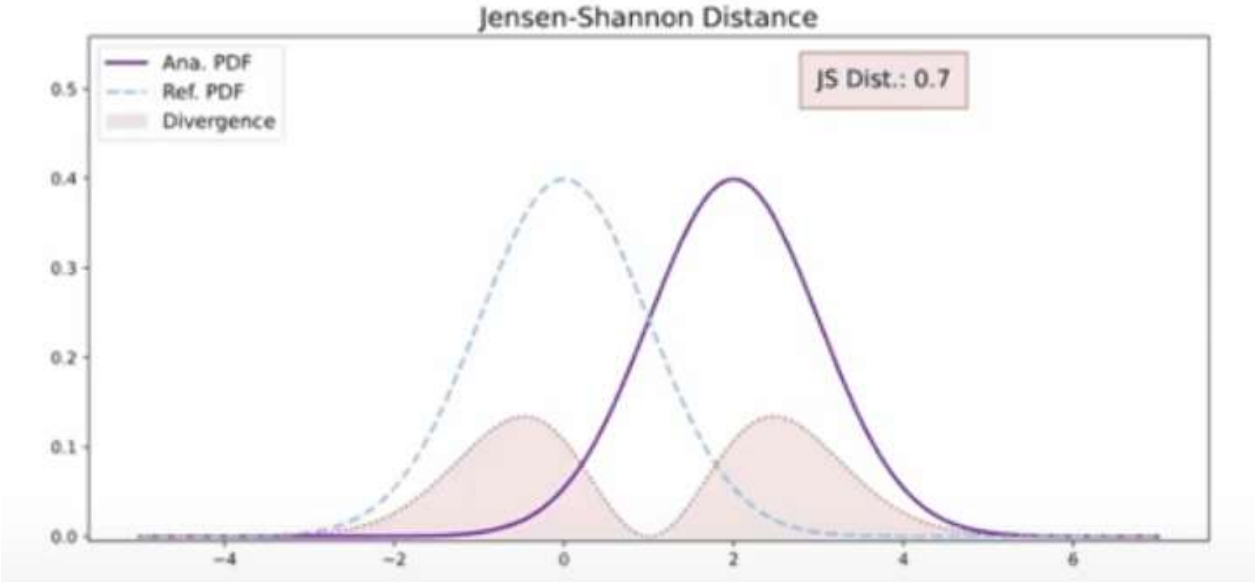
Comparison of two distributions with distance metrics

- Wasserstein distance
  - How much work to change one distribution into the other
  - Sensitive to outliers
  - Larger is distributions are more different



Comparison of two distributions with distance metrics

- Jensen-Shannon distance
  - The amount of overlap between distributions
  - Smaller = distributions are more different



Tests overview

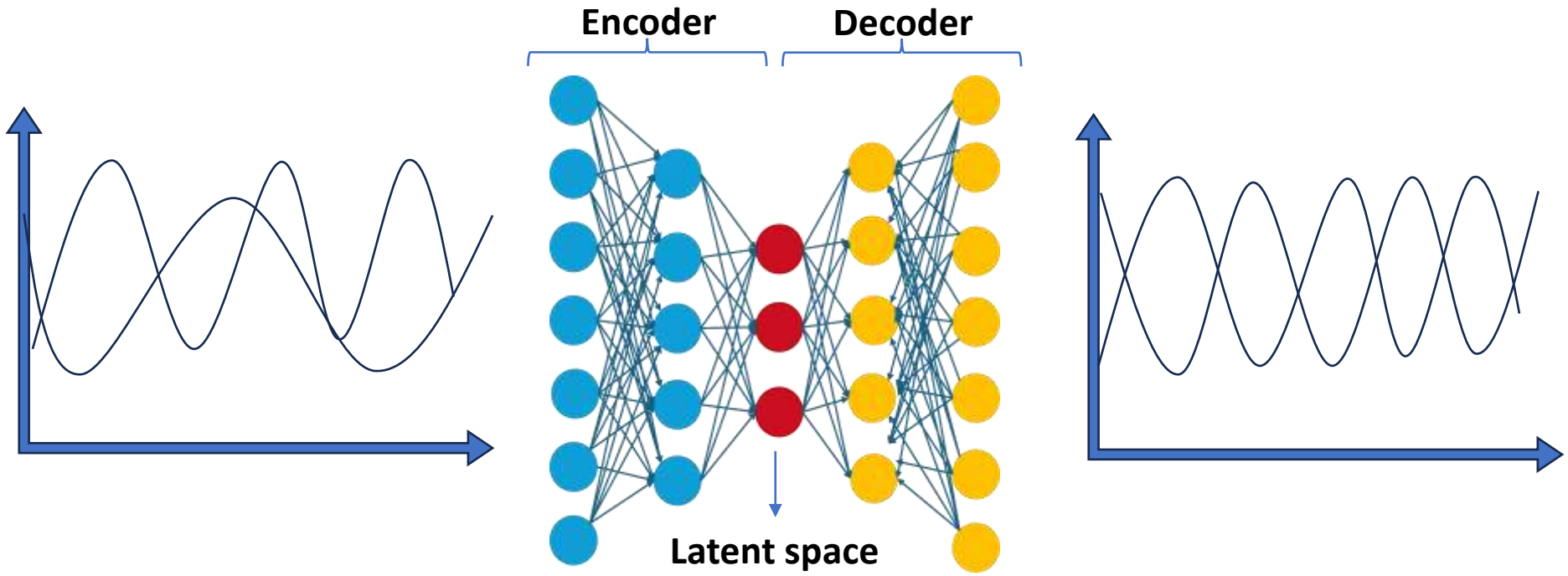
- Statistical = More sensitive in smaller distributions
- Distribution = Better for larger distributions
- Evidently default usage:

	<= 1000 objects	> 1000 objects
Numerical	KS	Wasserstein
Categorical	Chi-squared	Jensen-Shannon

Monitoring

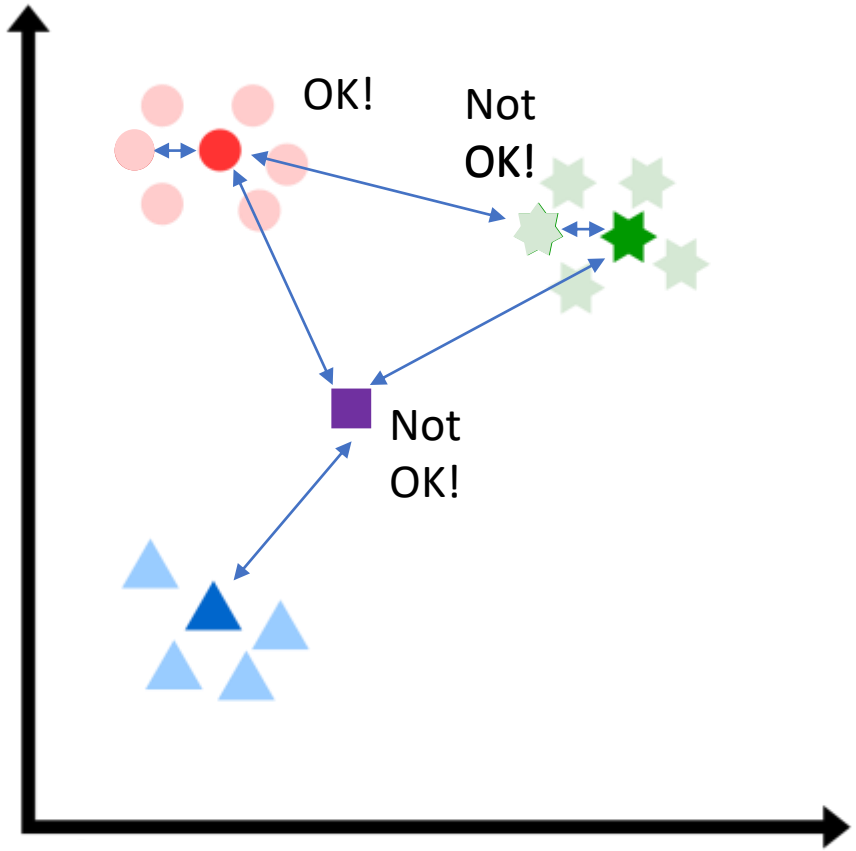
- What is drift
- Monitoring tools
- Data quality
- Model quality
- Statistical tests
- **Using machine learning models**
- Other types of drift

Autoencoder reproduces its input



Kmeans

Current Cluster =



Drift!

Monitoring

- What is drift
- Monitoring tools
- Data quality
- Model quality
- Statistical tests
- Using machine learning models
- **Other types of drift**

Monitoring - Other types of drift detection

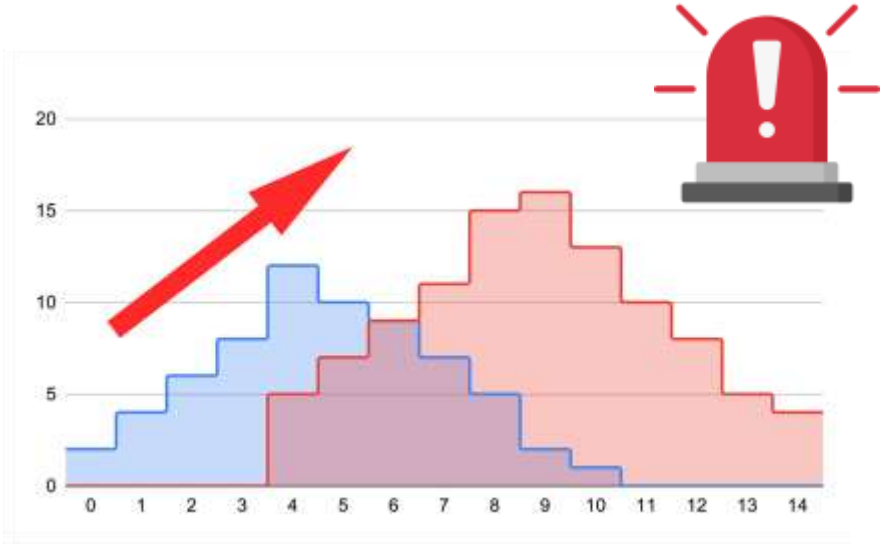
- Business metrics / Key Performance Indicators
- Fairness
- Resource consumption
  - Memory usage
  - Processing power
- ...

Closing the loop

- **When?**
- How?

Closing the loop – When?

- Scheduled: daily, weekly, monthly...
- When drift is detected

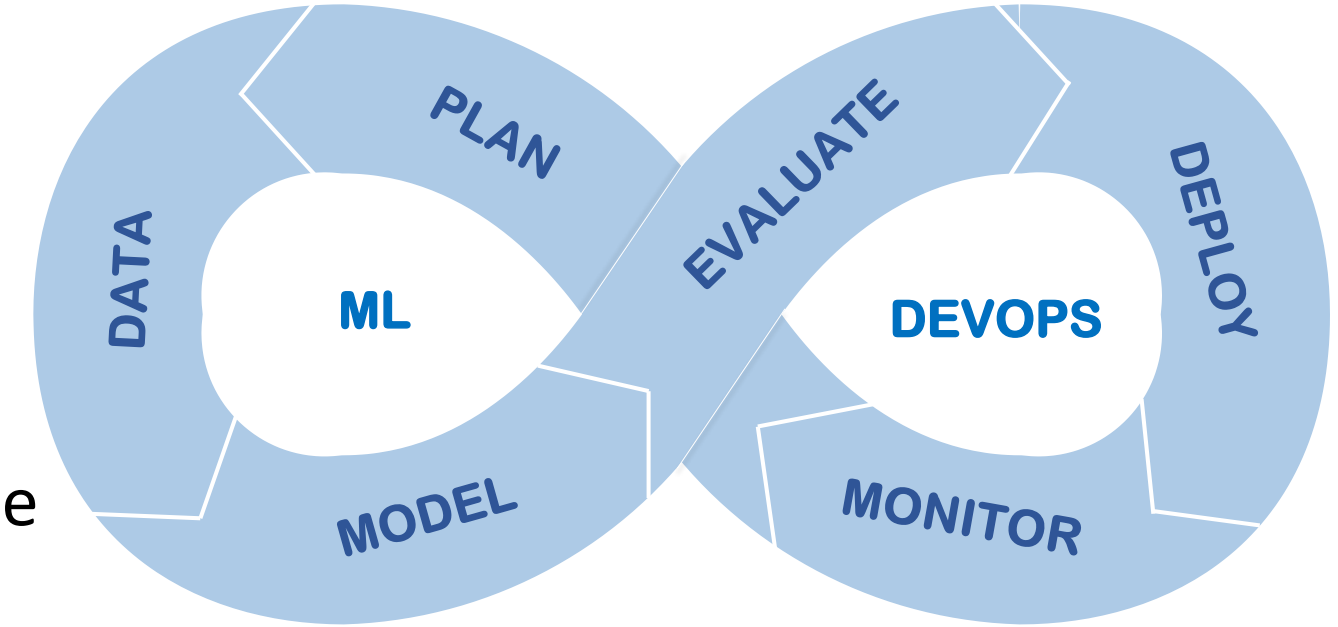


Closing the loop

- When?
- **How?**

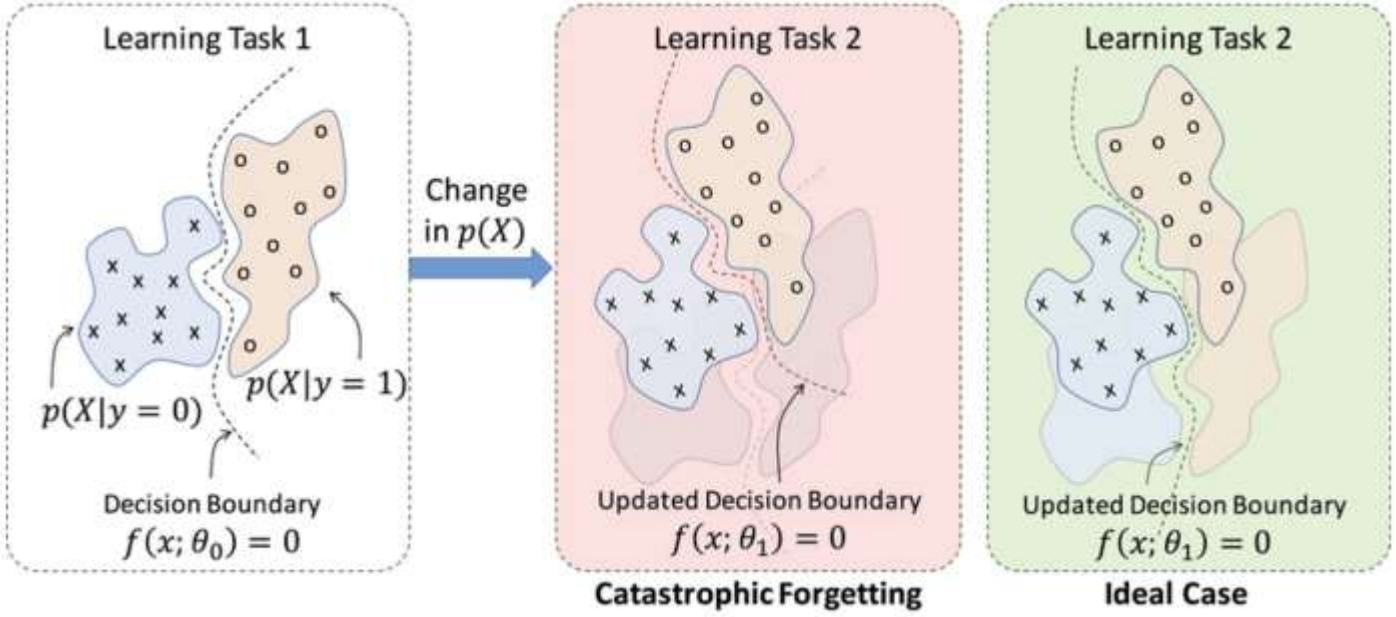
Creating a new model

- Start from scratch:
  - Plan new model
  - Get new data
  - Create new model
  - …
- Problem → takes a long time



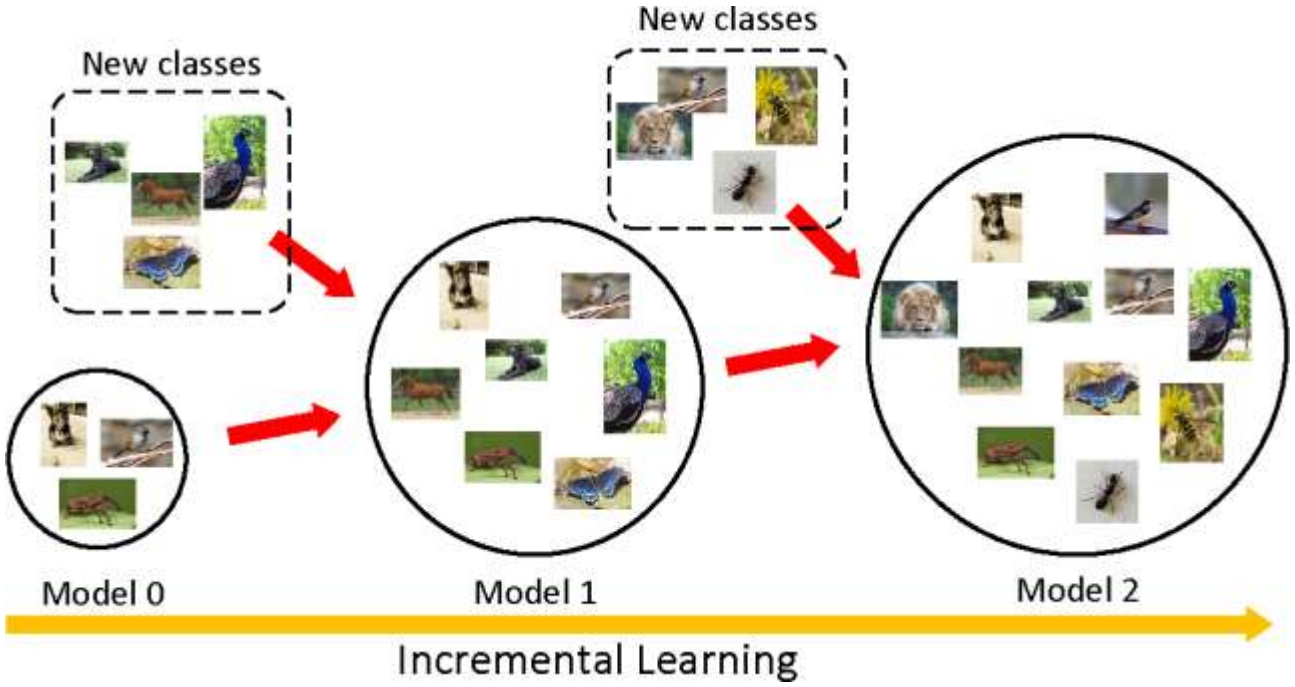
Lazy learning

- Train/finetune the old model on new data
- Fast option
- Problem → catastrophic forgetting



Incremental learning

- Training method without catastrophic forgetting
- Tradeoff → forgetting old data vs. learning new data



Ensemble method

- train multiple models and use the most relevant one