



# Ingénierie IA/ML

## Gilles Vanwormhoudt

# Plan

Introduction

Gestion des données

Ingénierie des caractéristiques

Entrainement et évaluation

Déploiement, surveillance et maintenance

# Introduction

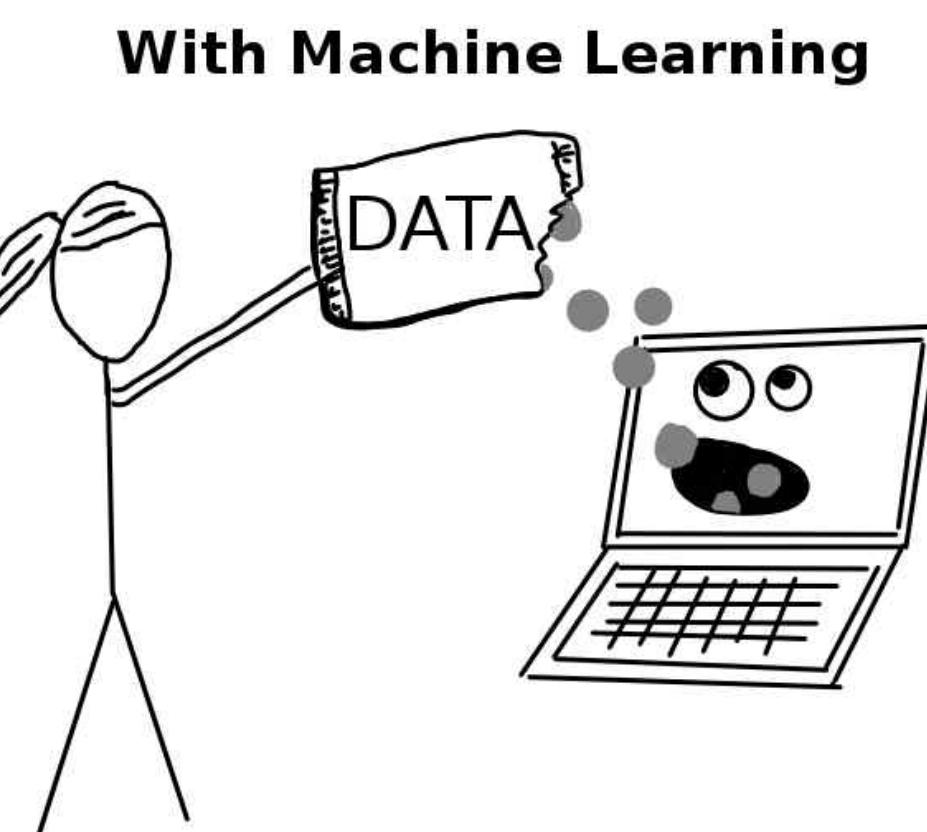
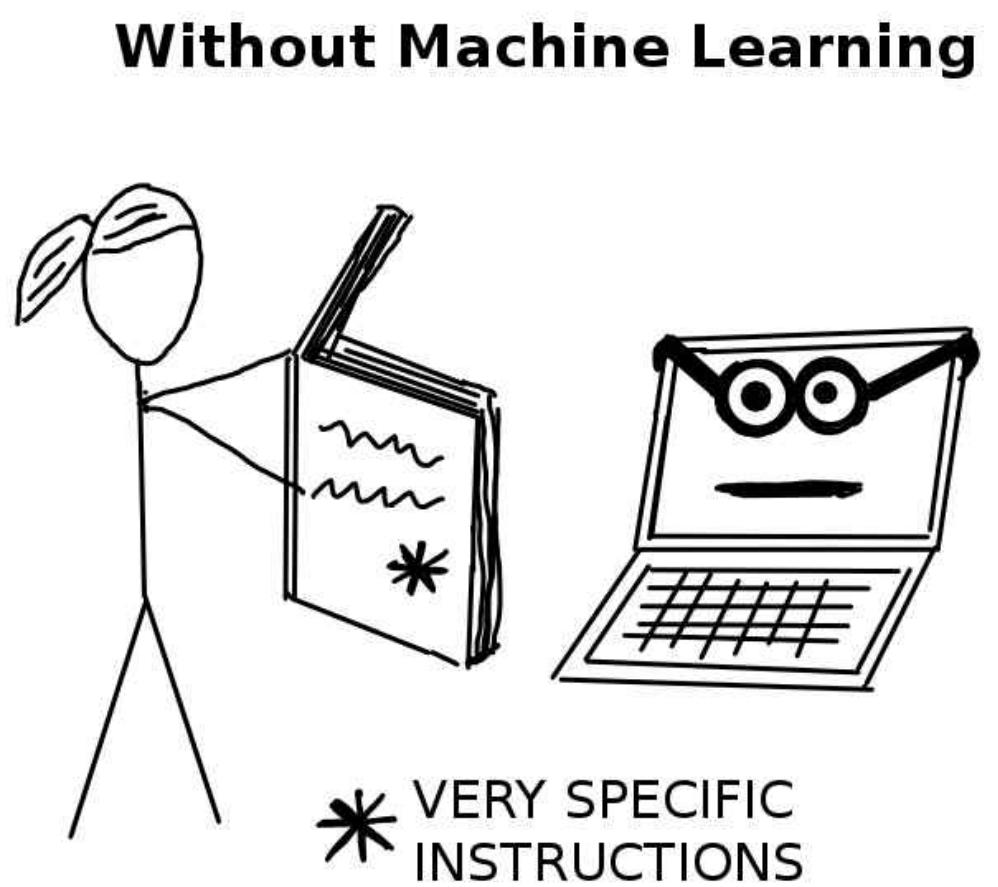
# Objectifs et périmètre du cours

- Ce cours s'intéresse à la mise en oeuvre de projets informatiques dont le cœur est basé sur du **Machine Learning**
  - Connaitre les différentes étapes d'un projet orienté ML
  - Connaitre les difficultés posées par ce type de projet et quelques techniques pour les résoudre
  - Traite des aspects implémentatoires (collection des données, préprocessing, testing, debugging, déploiement, contrôle, maintenance)
- Ne traite pas des aspects théoriques (types de machine learning, principales familles d'approches, ...)

# Projet ML

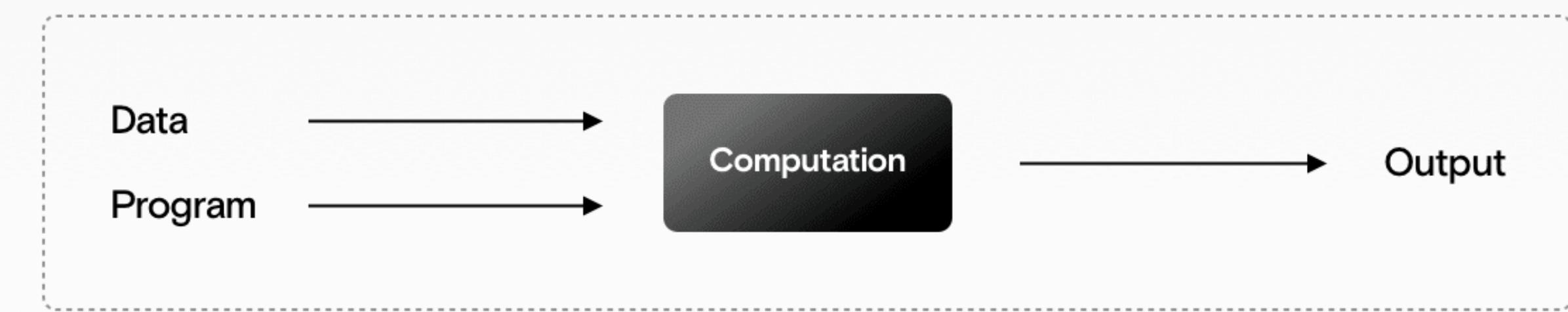
- Fait appel à des principes, techniques et des outils relevant du ML et de l'ingénierie logicielle pour concevoir et construire un système de calcul complexe
- Englobe de nombreuses étapes de la collection des données, de l'entraînement du modèle jusqu'au déploiement pour une utilisation par des utilisateurs ou des clients
- Nature différente par rapport à un projet logiciel classique (comportement déterministe)
  - Contient des étapes spécifiques
  - Incorporation de modèle qui peuvent se dégrader avec le temps ou avoir un comportement erroné
  - Très dirigé par les données -> changement dans la donnée (nouvelle distribution)
  - Prévoir et prévenir ces failles tout au moins les détecter et les gérer quand elles se produisent

# Projet classique vs Projet ML



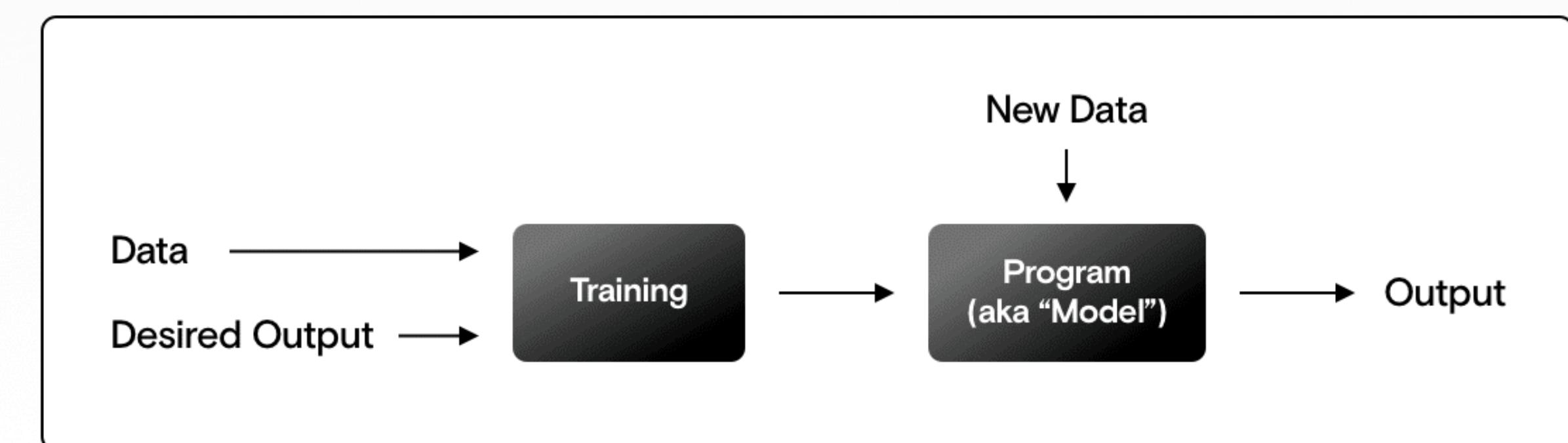
## Traditional Programming

Developers write rules (program) that produce an output.



## Machine Learning

Developers write a training algorithm, that finds rules, which produce the desired output.



# Quand utiliser le ML ?

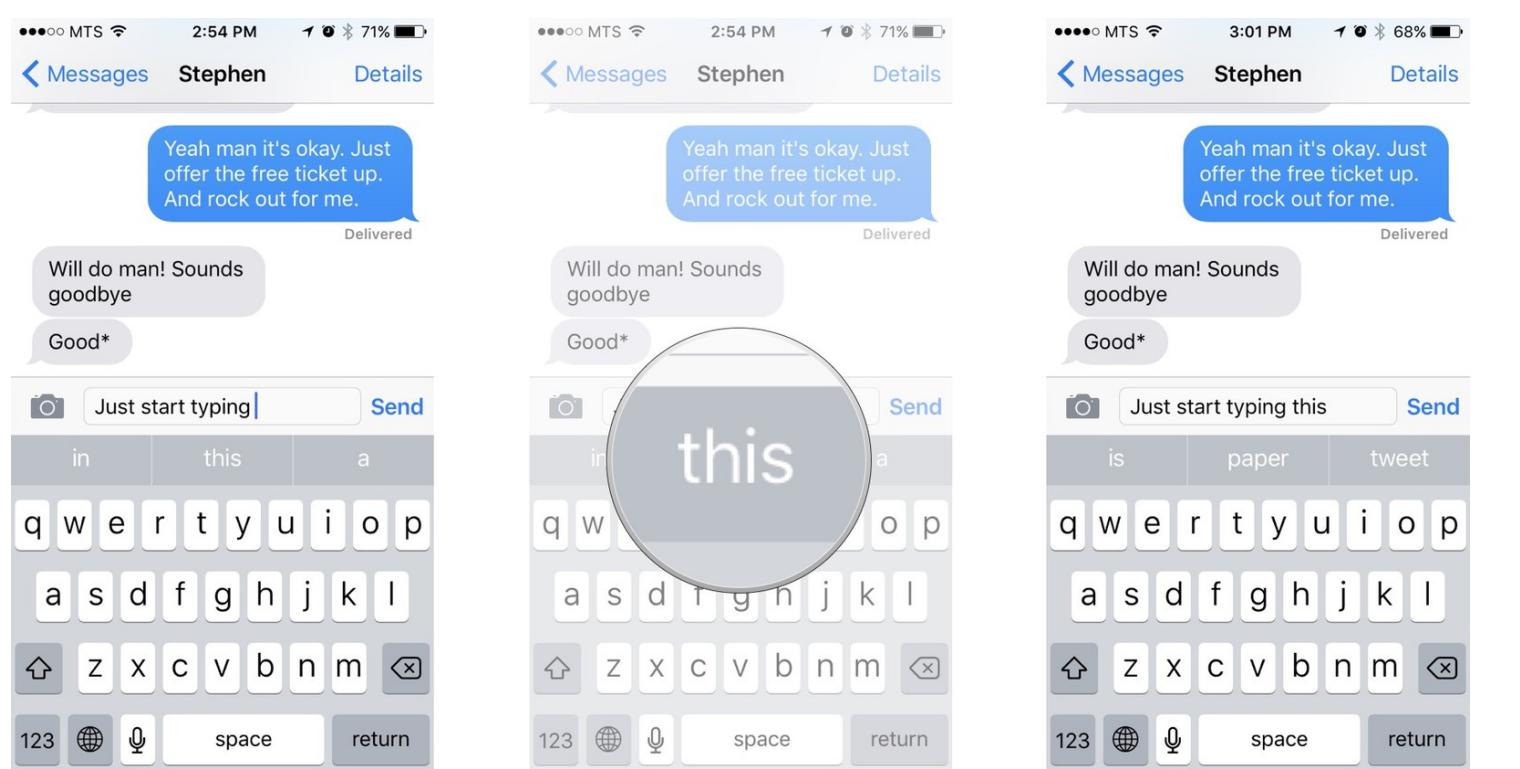
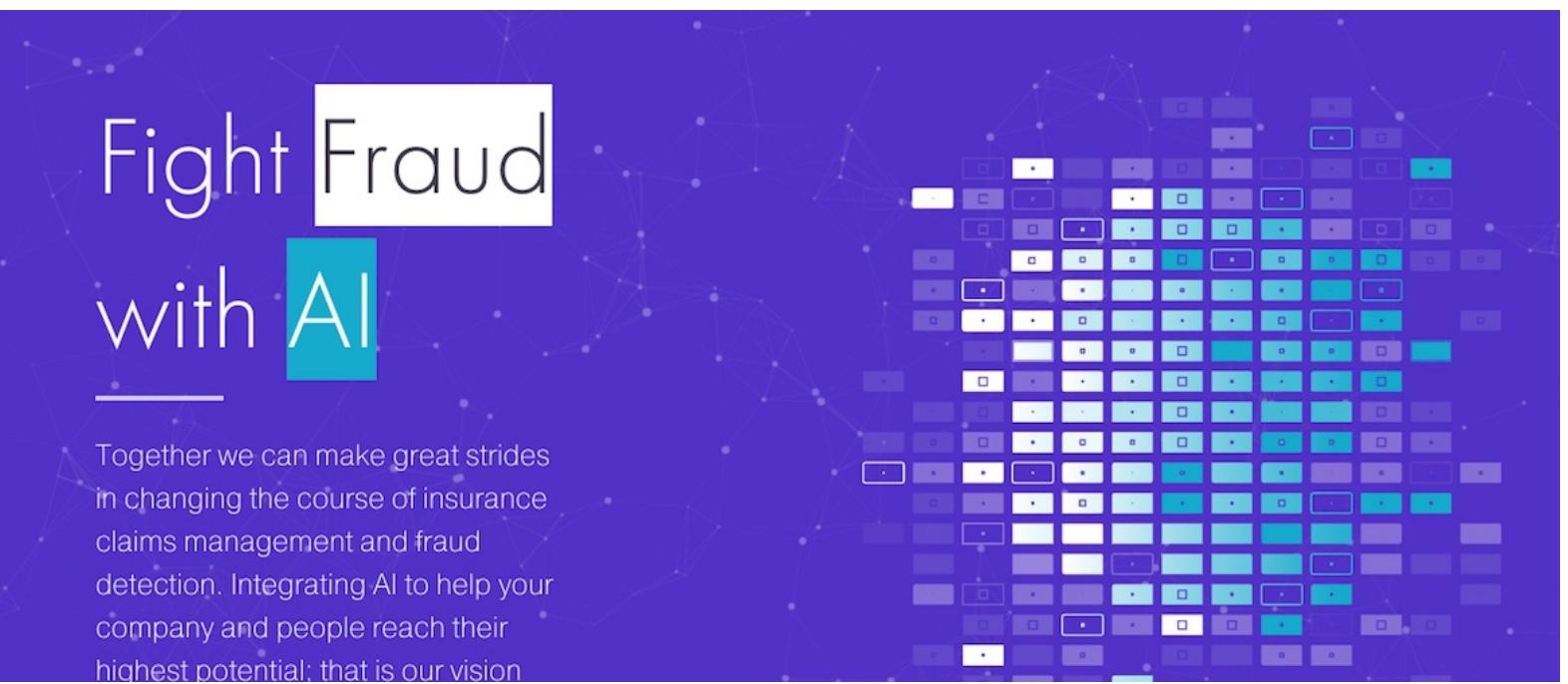
- Quand le problème est trop complexe pour être codé (trop de règles)
  - ▶ détection de spam, analyse de sentiments, ...
- Quand le problème change constamment nécessitant une mise à jour du code
- Quand le problème fait appel à la perception
- Quand le problème est une phénomène difficile à étudier mais pour lequel possède des observations (découverte de pattern, floue)

# Quand ne pas utiliser le ML ?

- Quand il faut être capable d'expliquer l'action ou la décision du système
- Quand le coût d'une erreur est très élevé voir intolérable
- Quand l'obtention de donnée appropriée est trop difficile ou impossible
- Quand vous pouvez résoudre le problème avec du logiciel à plus faible cout
- Quand une ou des heuristiques fonctionneraient suffisamment bien
- Quand vous pouvez résoudre le problème par une recherche pas trop couteuse dans une table de correspondance (entrée/sortie)

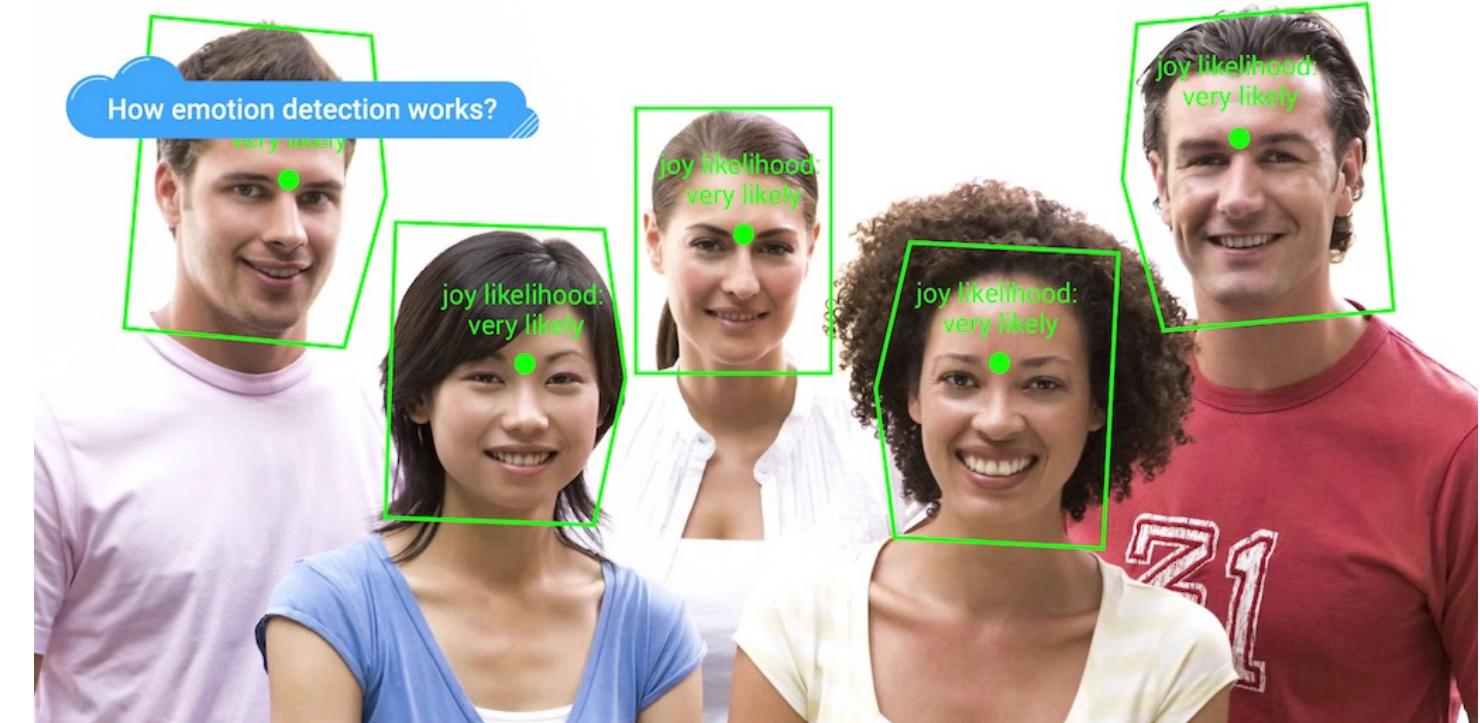
# Ce que peut faire un modèle

- **Automatiser** (par exemple, en prenant des mesures au nom de l'utilisateur ou en démarrant ou arrêtant une activité spécifique sur un serveur),
- **Alerter ou inviter** (par exemple, en demandant à l'utilisateur si une action doit être entreprise ou en demandant à un administrateur système si le trafic semble suspect),
- **Organiser**, en présentant un ensemble d'éléments dans un ordre qui pourrait être utile à un utilisateur (par exemple, en triant des images ou des documents dans l'ordre de similarité avec une requête ou selon les préférences de l'utilisateur),
- **Annoter** (par exemple, en ajoutant des annotations contextuelles aux informations affichées, ou en mettant en évidence, dans un texte, des phrases pertinentes pour la tâche de l'utilisateur),
- **Extraire** (par exemple, en détectant de petits éléments d'information pertinents dans une entrée plus importante, comme des entités nommées dans le texte : noms propres, entreprises ou lieux),
- **Recommander** (par exemple, en détectant et en montrant à un utilisateur des éléments très pertinents dans une grande collection, en fonction du contenu de l'élément ou de la réaction de l'utilisateur aux recommandations antérieures),



# Ce que peut faire un modèle

- **Classer** (par exemple, en répartissant des exemples d'entrée dans un ou plusieurs groupes d'un ensemble prédéfini de groupes aux noms distincts),
- **Quantifier** (par exemple, en attribuant un nombre, comme un prix, à un objet, comme une maison),
- **Synthétiser** (par exemple, en générant un nouveau texte, une image, un son ou un autre objet similaire aux objets d'une collection),
- **Répondre à une question explicite** (par exemple, "Ce texte décrit-il cette image ?" ou "Ces deux images sont-elles similaires ?"),
- **Transformer son entrée** (par exemple, en réduisant sa dimensionnalité à des fins de visualisation, en paraphrasant un long texte pour en faire un court résumé, en traduisant une phrase dans une autre langue, ou en augmentant une image en lui appliquant un filtre),
- **Déetecter** une nouveauté ou une anomalie.



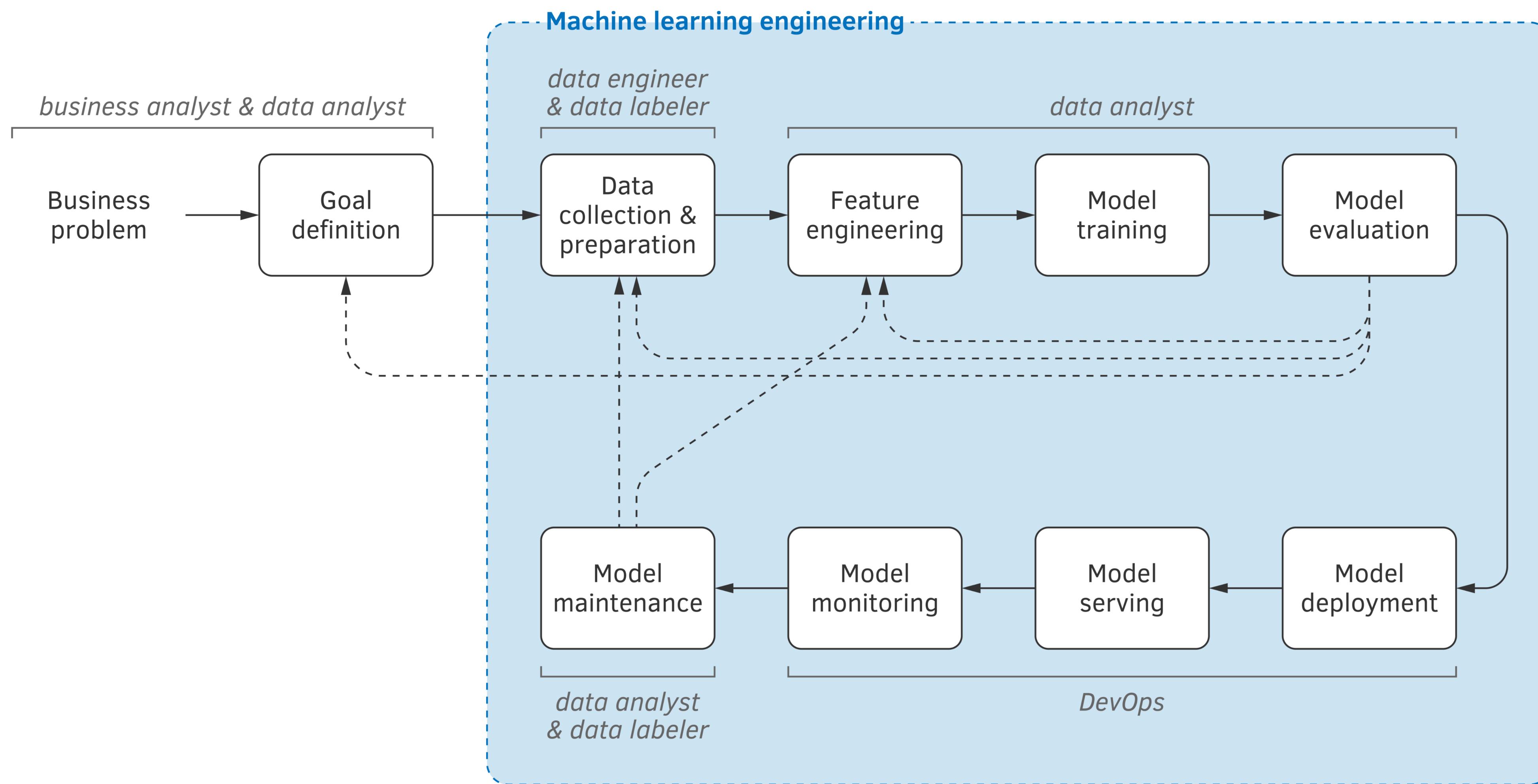
TEXT PROMPT  
an armchair in the shape of an avocado [...]



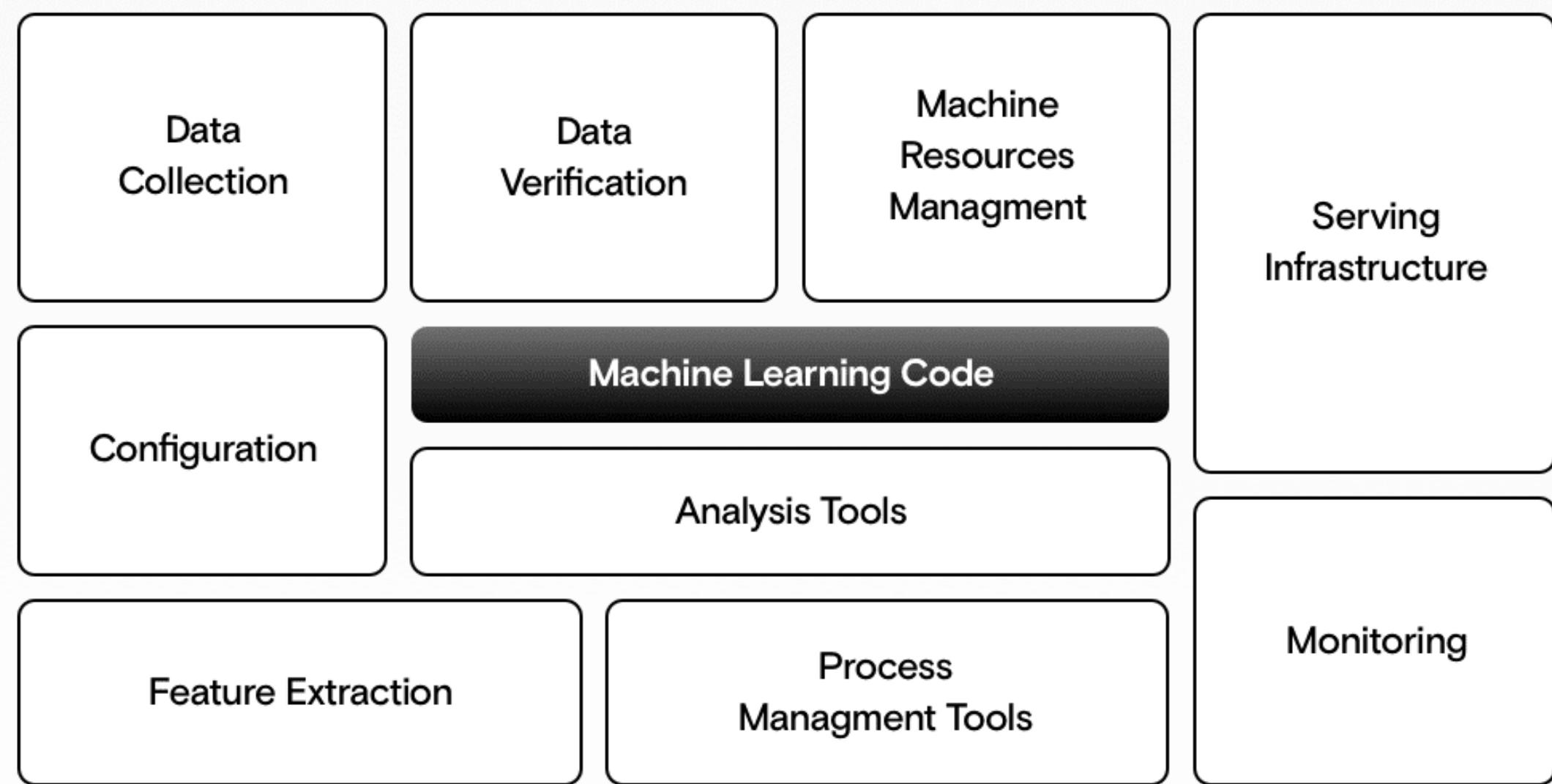
AI-GENERATED IMAGES



# Projet ML : Vue d'ensemble



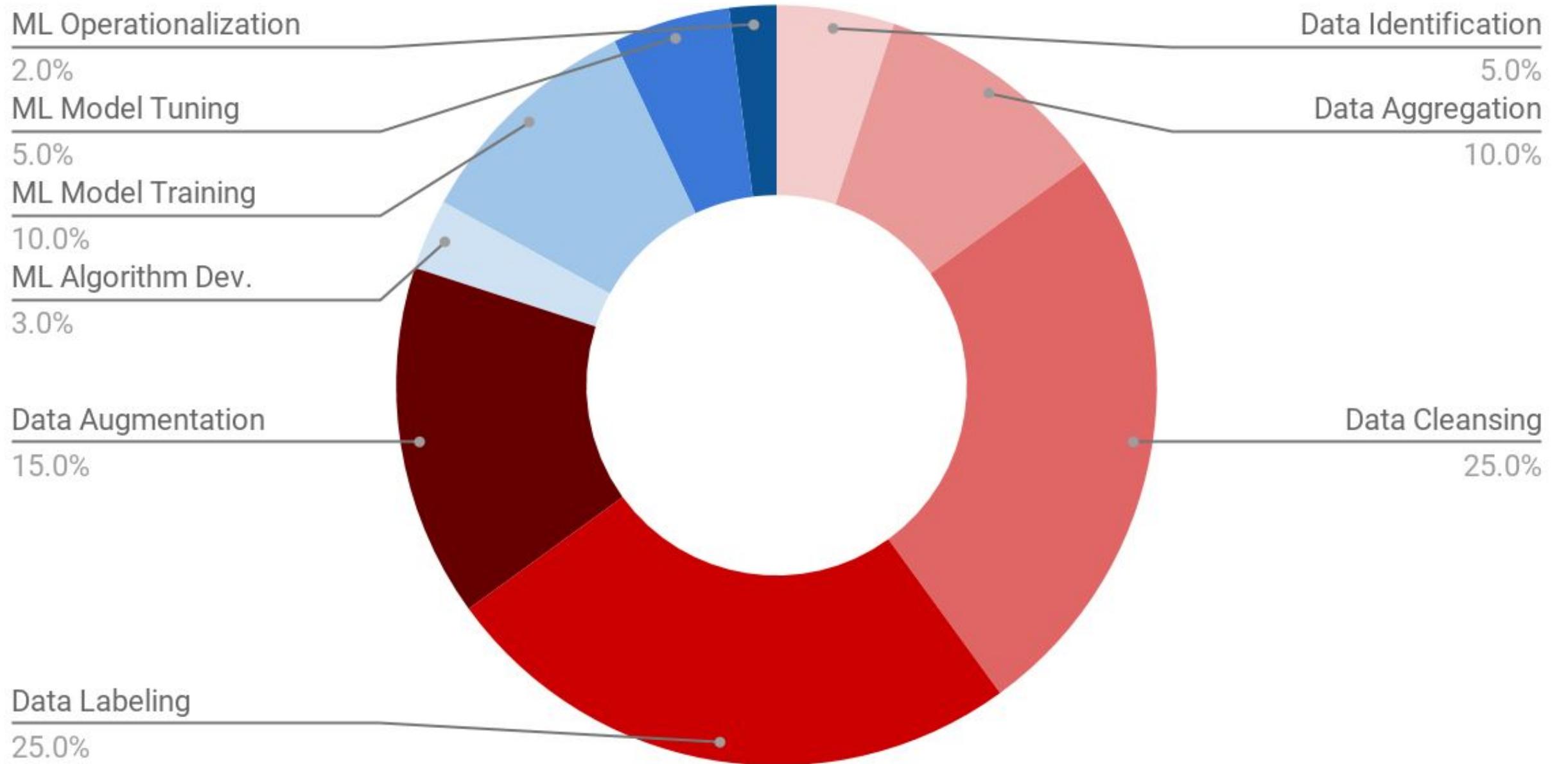
# Project ML : Vue d'ensemble



ML code is only a small fraction of real-world ML solutions (black box in the middle). The required surrounding infrastructure is vast and complex.

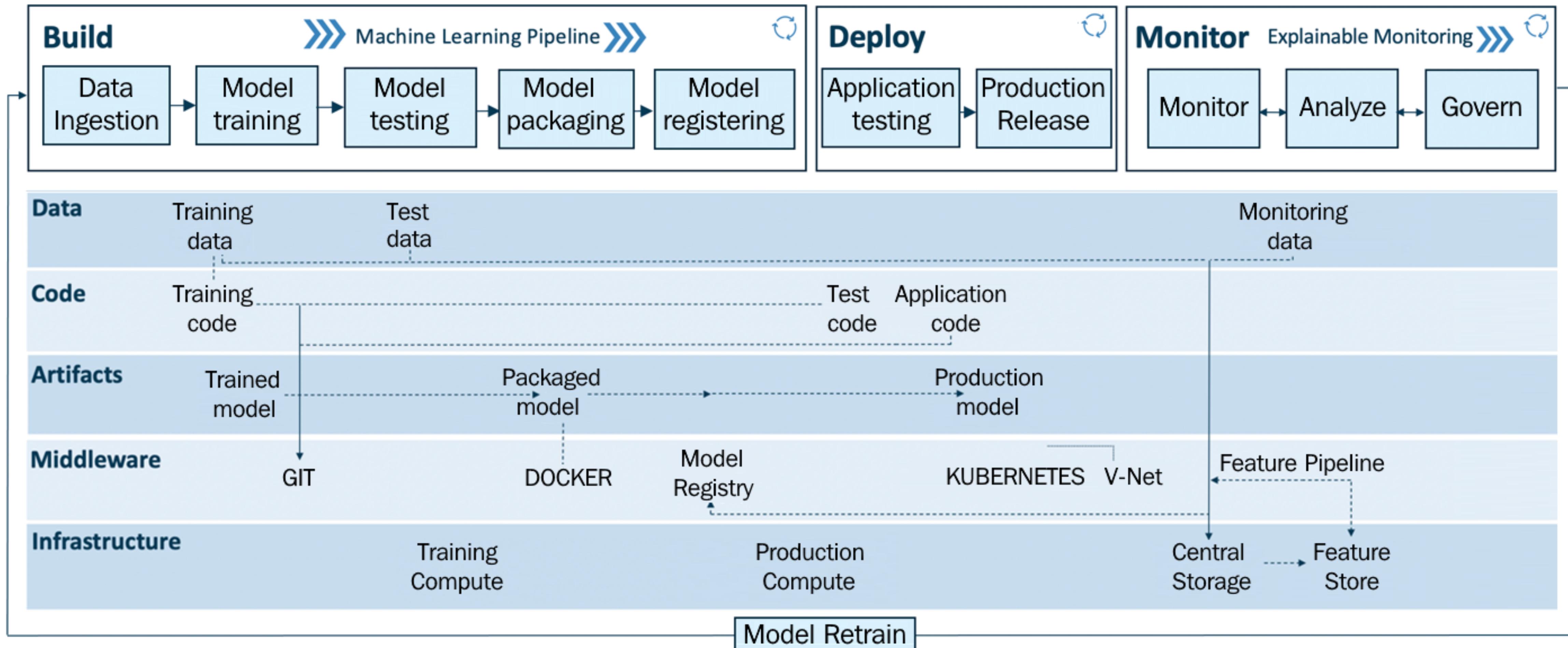
Percentage of Time Allocated to Machine Learning Project Tasks

Source: Cognilytica

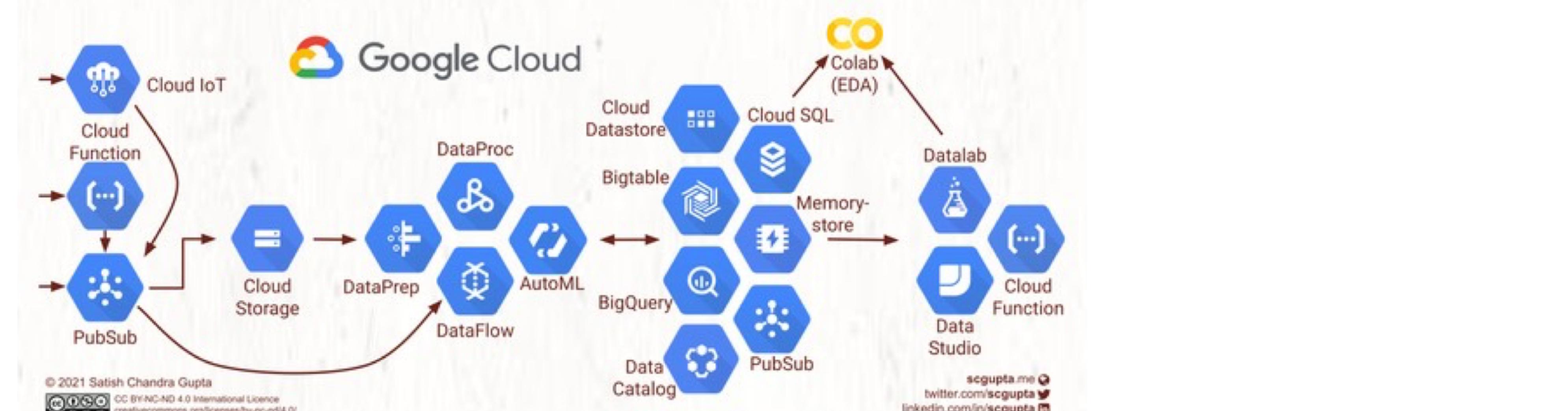
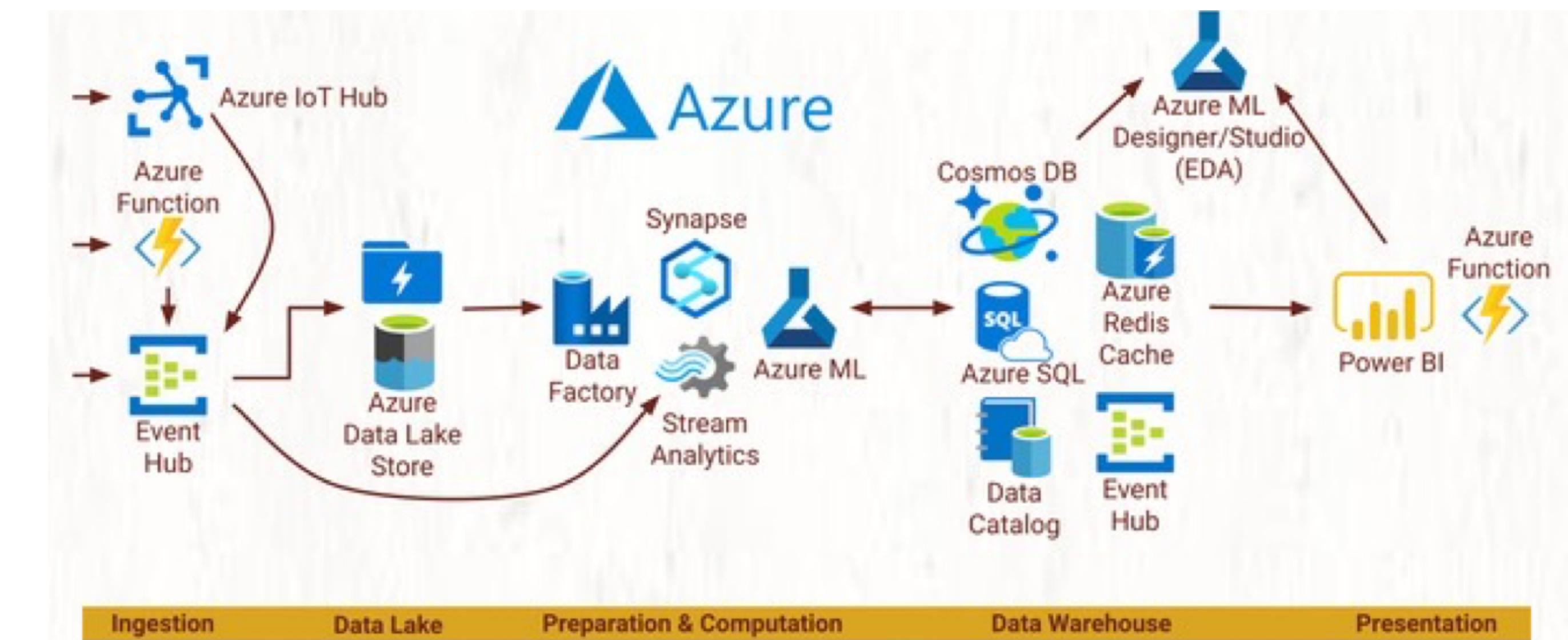
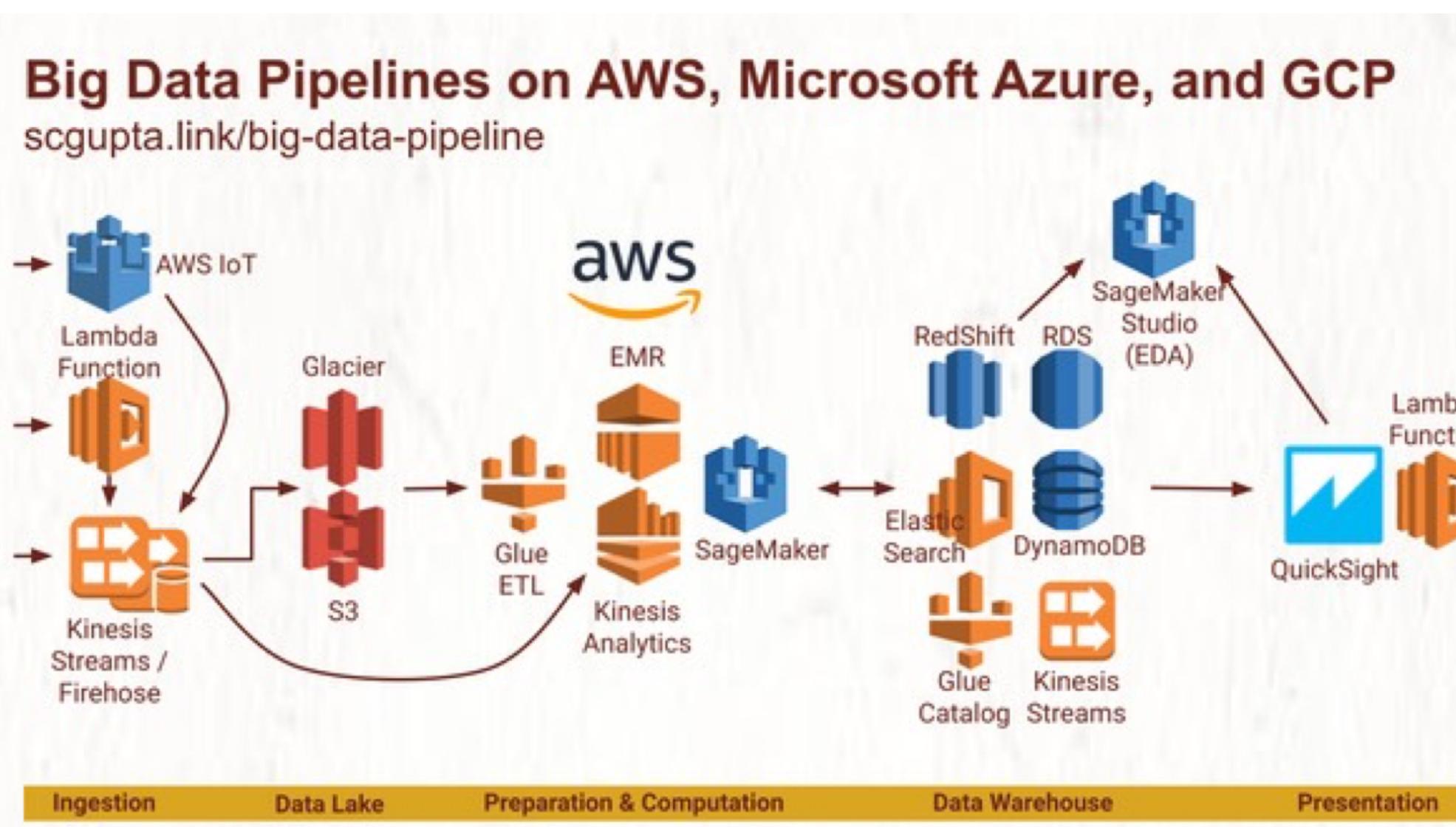


# Projet ML: Vue d'ensemble

## MLOps Workflow

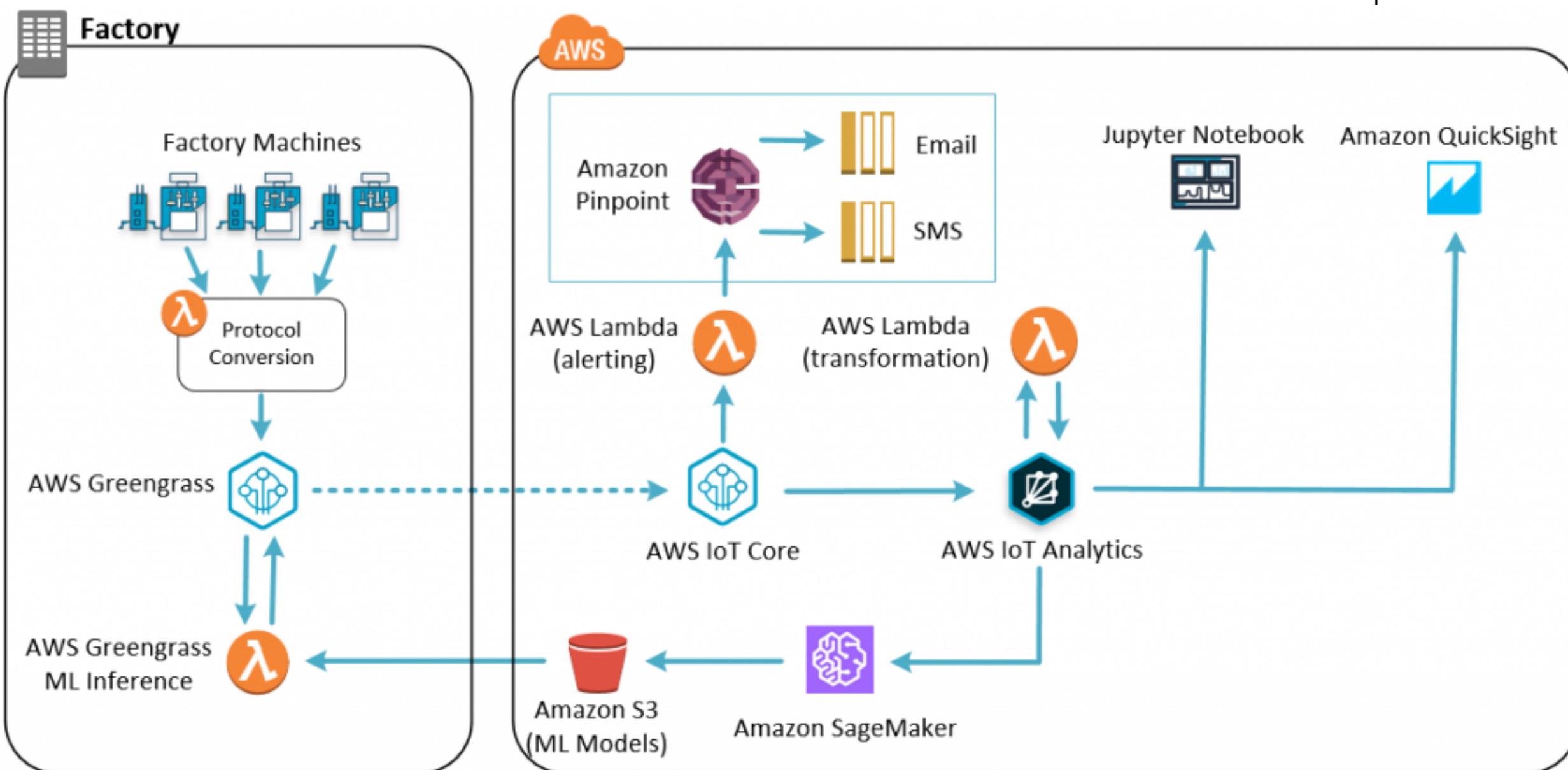


# Project ML: Instantiation

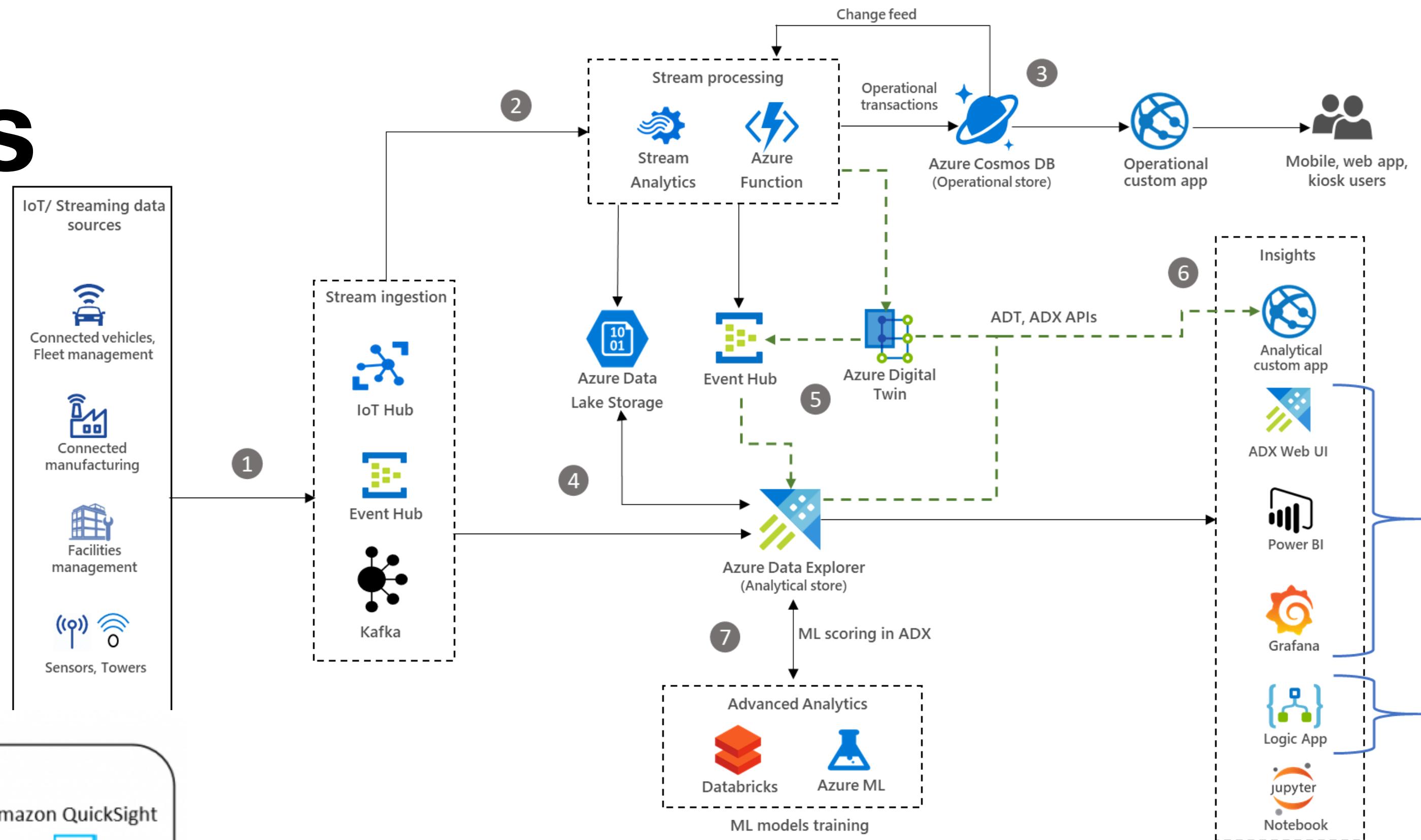


# Projet ML: Exemples

Maintenance preventive



Internet des objets



# Au début du projet

- Comprendre l'objectif métier / business et le transformer en un projet d'ingénierie
- Spécifier:
  - Les données en entrées et les sorties attendus
  - Le critère d'acceptabilité / inacceptabilité du comportement du système :
    - 80 % des cas l'algorithme ne se trompe pas, 95 % des cas fournit une bonne prédiction
- Evaluer:
  - Difficulté du problème (bibliothèque existante ou pas, nouveau problème ou pas - ressemblance avec des tâches connues : classification, regression, génération, ...)
  - Cout de la donnée (générer automatique, annotation manuelle, volume requis)
  - Le besoin de précision (critère d'acceptabilité)

# Difficultés des projets ML<sup>1</sup>

- Manque de compétences et de collaborateurs expérimentés : pas uniquement des data-scientists, il faut aussi des ingénieurs ML, des MLOps, des experts du domaines
- Manque de soutien de la hiérarchie (problèmes de compréhension des objectifs, des contraintes et des risques : scientifique vs non-scientifique)
- Incertitude / difficultés à atteindre les résultats, perte de temps et d'argent, promotion de l'IA dans les média
- Manque d'infrastructure adéquates à la fois pour la donnée et les traitements (outils/moteur spécifique pour les données (qualité, transformation), matériel (GPU), architecture (clusters),)

# Difficultés des projets ML<sup>2</sup>

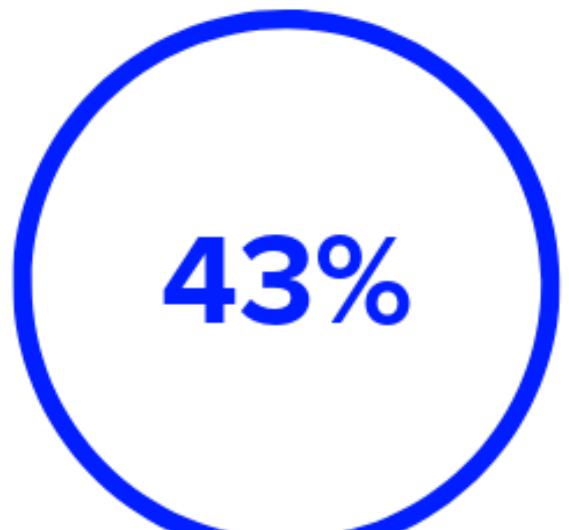
- Défi de l'étiquetage des données (obtenir une qualité et une quantité suffisante, outsourcing)
- Contraintes de l'organisation (silo, manque de collaboration, incompréhension, cloisonnement des services)
- Problème d'alignement des objectifs et des équipes techniques et business
- Progression non-linéaire (diminution des progrès, remise en cause des modèles avec de nouvelles données)
- Infaisabilité du projet (objectif non réaliste, trop couteux, nécessite trop de temps)
- Nombreuses inconnues (qualités et quantité de données requises, choix des bonnes caractéristiques, choix des algorithmes et des hyperparamètres, durée d'entraînement)
- Reproductibilité des résultats

# Gestion des données

# Principales tâches

- Récupérer la donnée
- Analyse de la donnée et de son cycle de vie
- Evaluation de la qualité
- Transformer la donnée pour l'améliorer
- Stocker la donnée

**Data quality is biggest ML challenge**



say poor data quality  
is biggest ML barrier



report lack of data  
availability as 2nd  
highest barrier



find it hard to hire  
data-scientist talent

#MLReadyData

Source: Refinitiv AI/ML Survey

# Collecter la donnée

- Déterminer
  - ▶ les sources (internes, externes)
  - ▶ la fréquence
  - ▶ le mode (push, pull, crawling)
  - ▶ le volume
  - ▶ statique vs dynamique
  - ▶ total vs partiel
- Impact sur les choix architecturaux pour le début de la pipeline



Figure 2 - Big Data Integration

# Questions sur la donnée

- Est-ce que les données sont facilement accessibles ?
  - Existent-elle ? Si oui, le sont-elles accessible physiquement, du point de vue éthique et économique, sont-elles sensibles, protégées, ... ?
- Est-ce que les données sont en nombre suffisants ?
  - Analyse des courbes d'apprentissage (voir figure), règles empiriques
- Est-ce que les données à dispositions sont exploitables ?
  - Trop de valeurs manquantes et de valeurs erronées, doublons fréquents, données périmés, ...
- Est-ce que les données sont compréhensibles ?
  - Etre capable d'interpréter les caractéristiques
- Est-ce que les données sont fiables ?
  - Fiabilité des étiquettes (travailleurs du Turc Mecanique), étiquette indirecte (click sur un lien) vs étiquette directe

# Bien connaître la donnée

- Analyser la donnée
  - ▶ comprendre la structure
  - ▶ comprendre les attributs: types, intervalle de valeurs, ...
  - ▶ comprendre le cycle de vie
    - partiel vs total (one-shot), ...
    - accroissement (temporel ou nb clients)
- Evaluer la qualité de la donnée
  - ▶ Outils de mesure
  - ▶ Outils de visualisation



# Qu'est ce qu'une bonne donnée

- Les bonnes données sont informatives
- Les bonnes données ont une bonne couverture
- De bonnes données reflètent des entrées réelles
- Les bonnes données sont impartiales
- Les bonnes données ont des étiquettes cohérentes
- Les bonnes données sont assez volumineuses

# Problèmes courants avec la donnée

- Coût élevé
- Mauvaise qualité
- Bruit
- Biais
- Exemples périmés
- Valeurs aberrantes
- Fuite de données



# Traitement des attributs manquants

- Supprimer les exemples (si ok pour sacrifice des données)
- Techniques d'imputations des données
  - ▶ Valeur numérique:

- moyenne
- médiane
- utilisation de régression

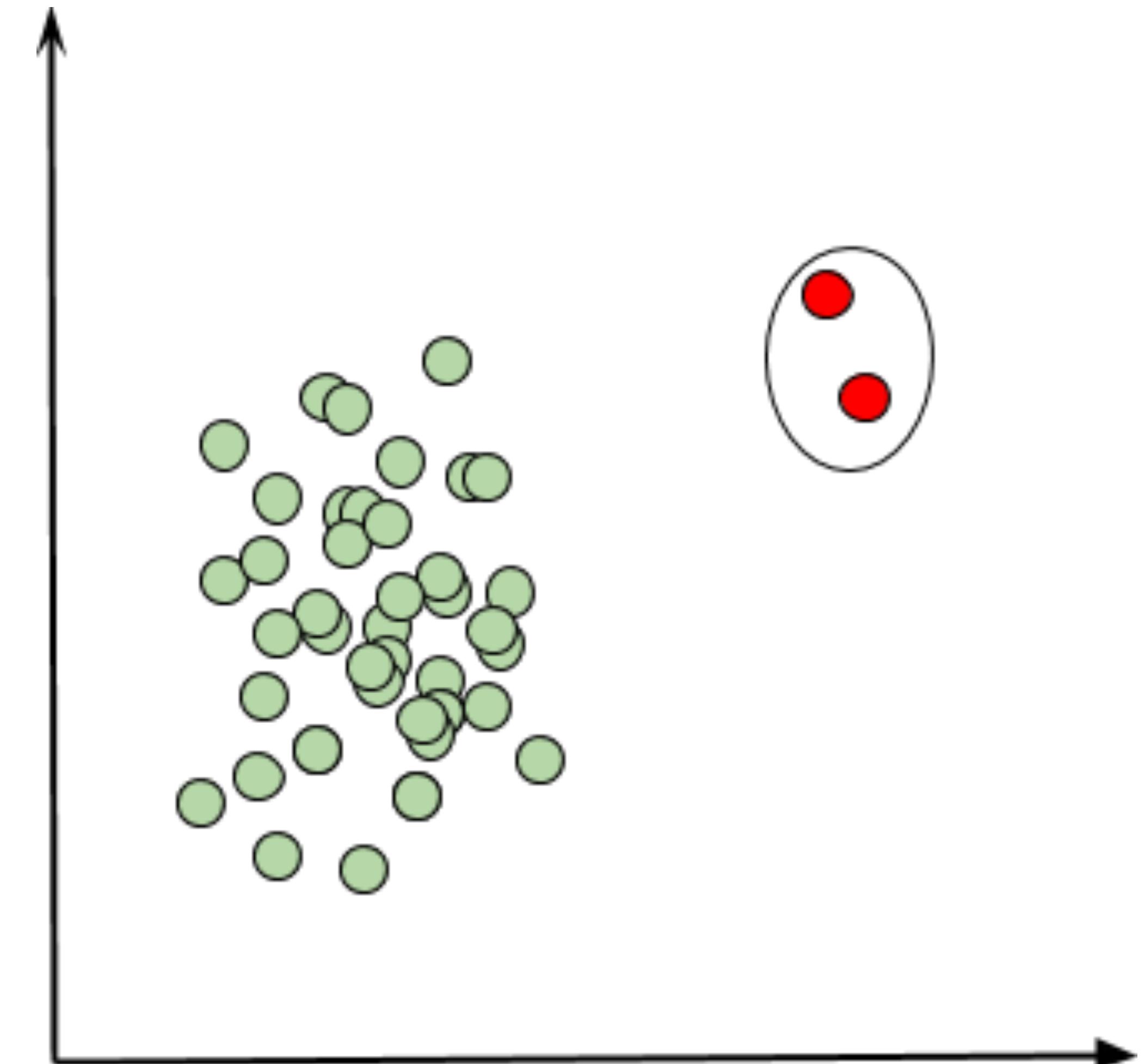
Row	Age	Weight	Height	Salary
1	18	70		35,000
2	43	65	175	26,900
3	34	87		76,500
4	21	66	187	94,800
5	65	60	169	19,000

$$\hat{Height} \leftarrow \frac{1}{3}(175 + 187 + 169) = 177$$

- ▶ Valeur catégorielle: ‘Unknown’

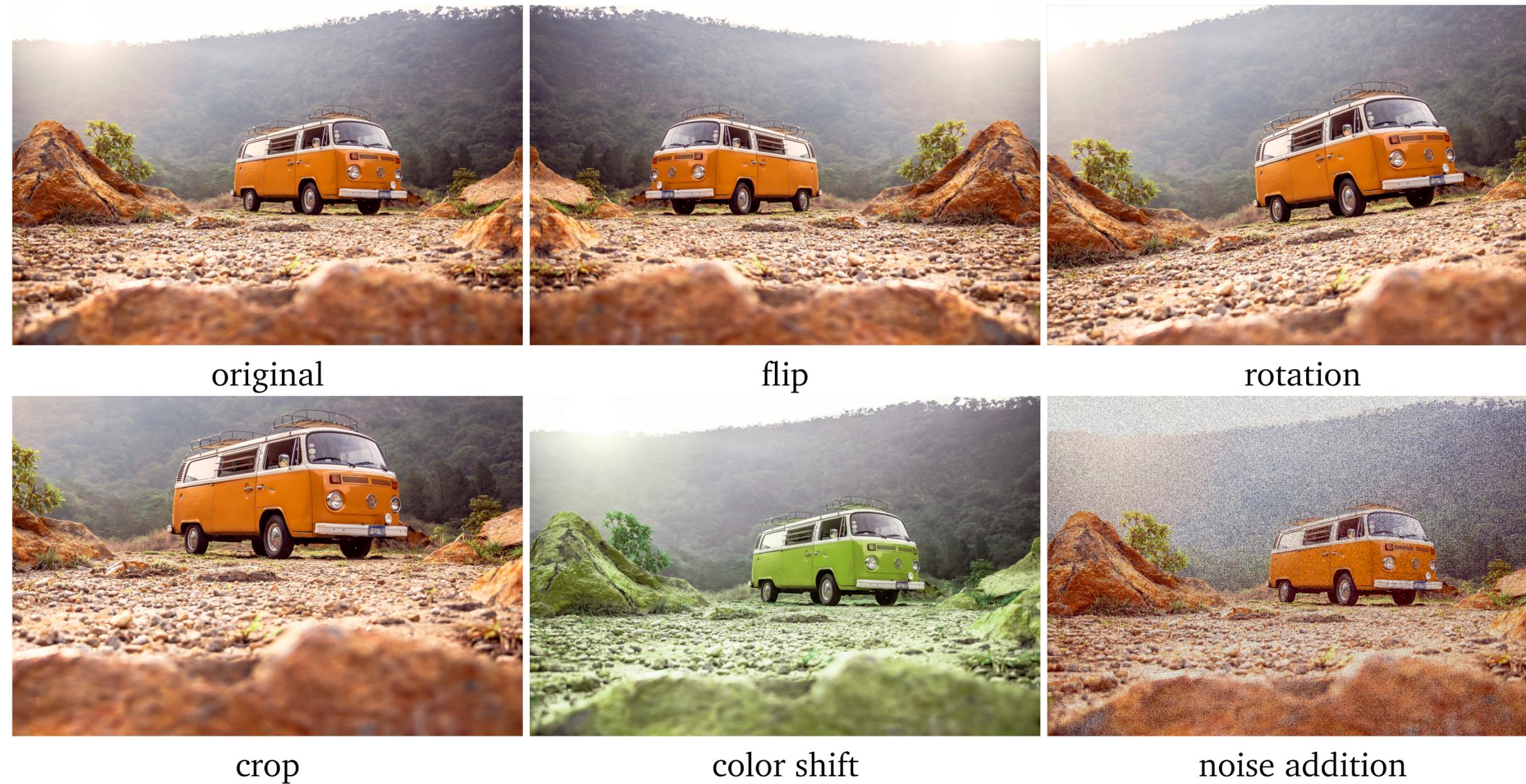
# Détection des Outliers

- Les outliers sont des valeurs extrêmes qui se situent très loin des autres observations
- Besoin de les détecter
  - Le jeu de données est petit
  - Si l'algorithme d'apprentissage utilisé y est sensible : Régression Logistique, AdaBoost, ...
- Exclusion à étudier mais pas systématique
- Techniques : similarité, autoencoders (erreur de reconstruction)

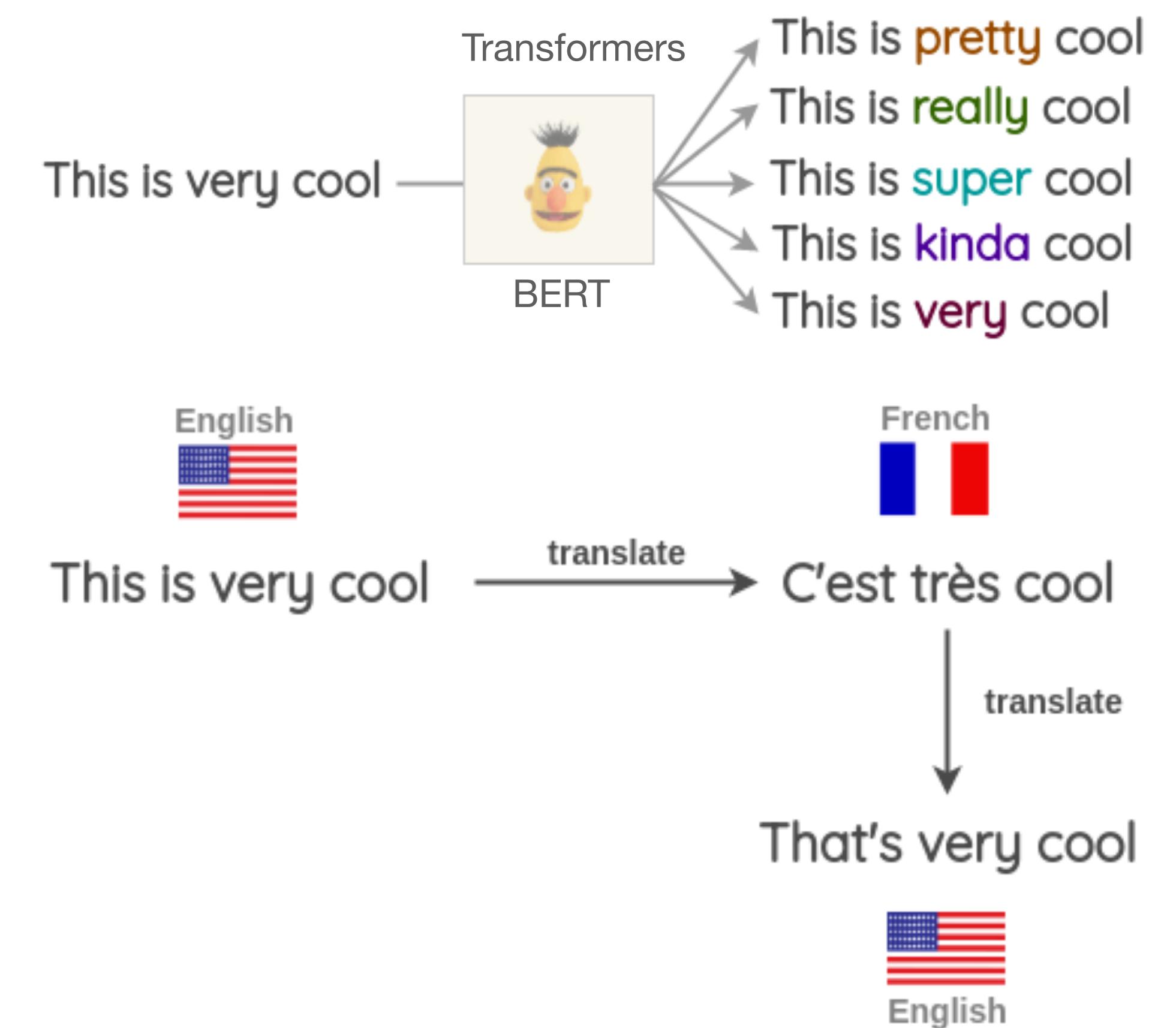


# Faible volume de données

- Techniques d'augmentation des données



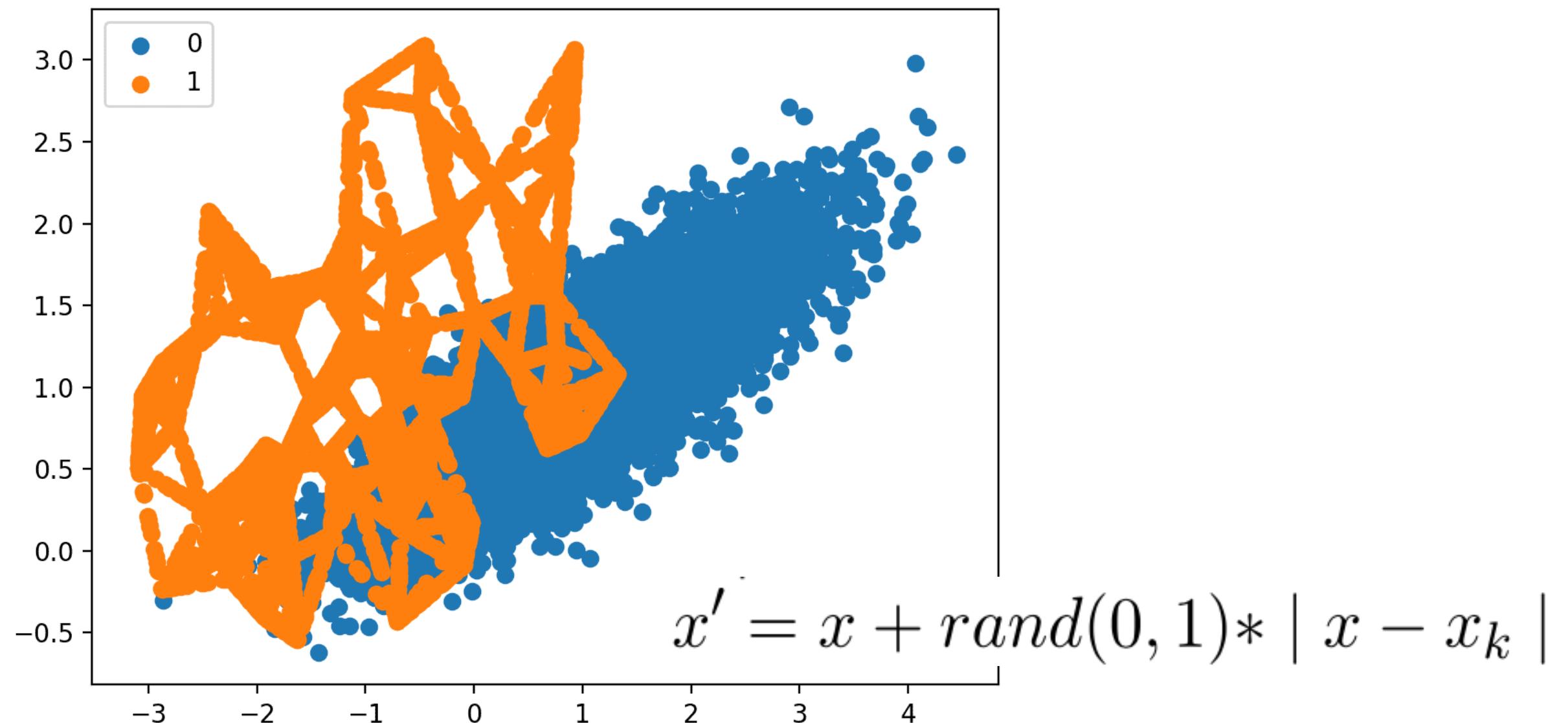
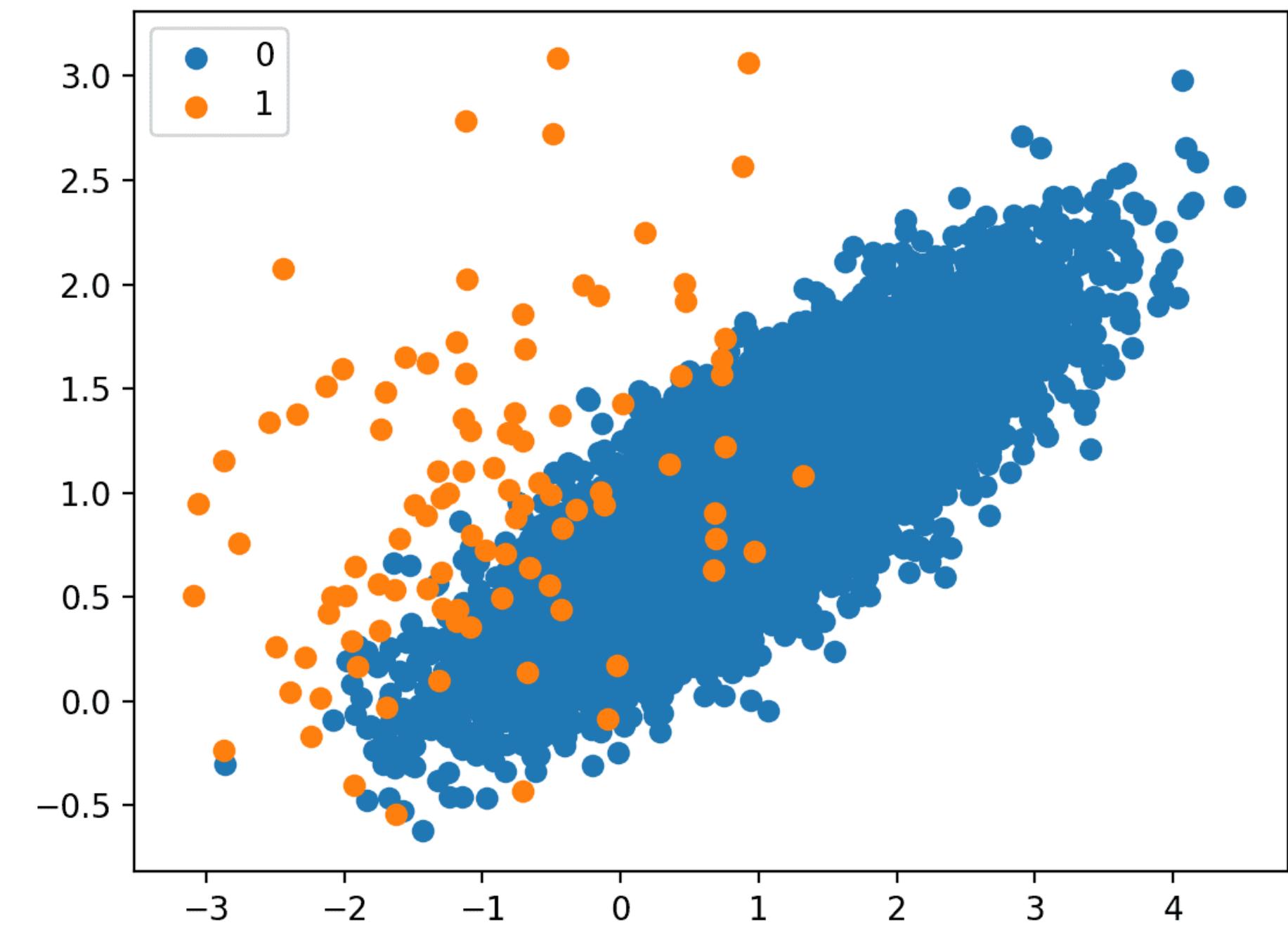
Opérations sur les images d'entraînement



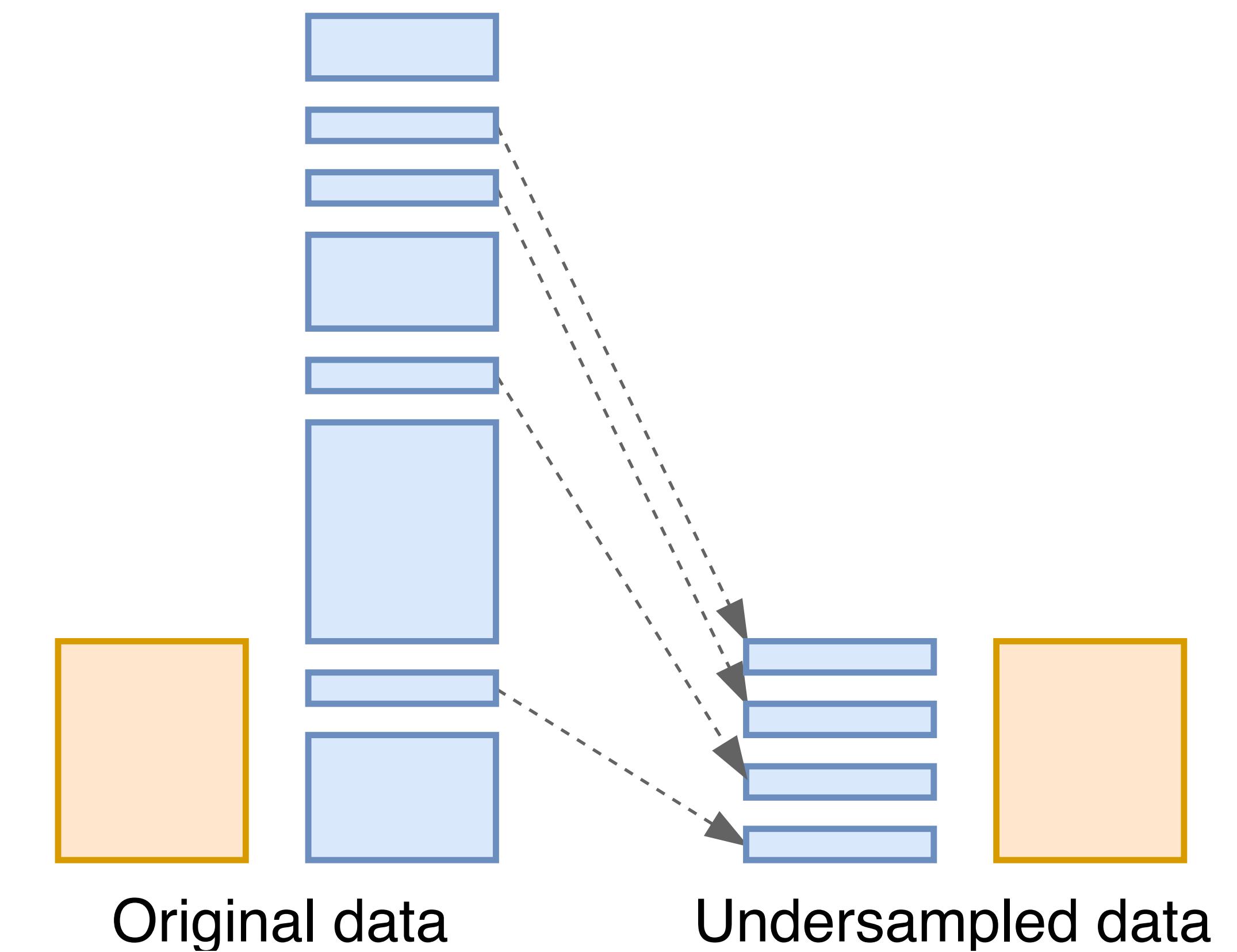
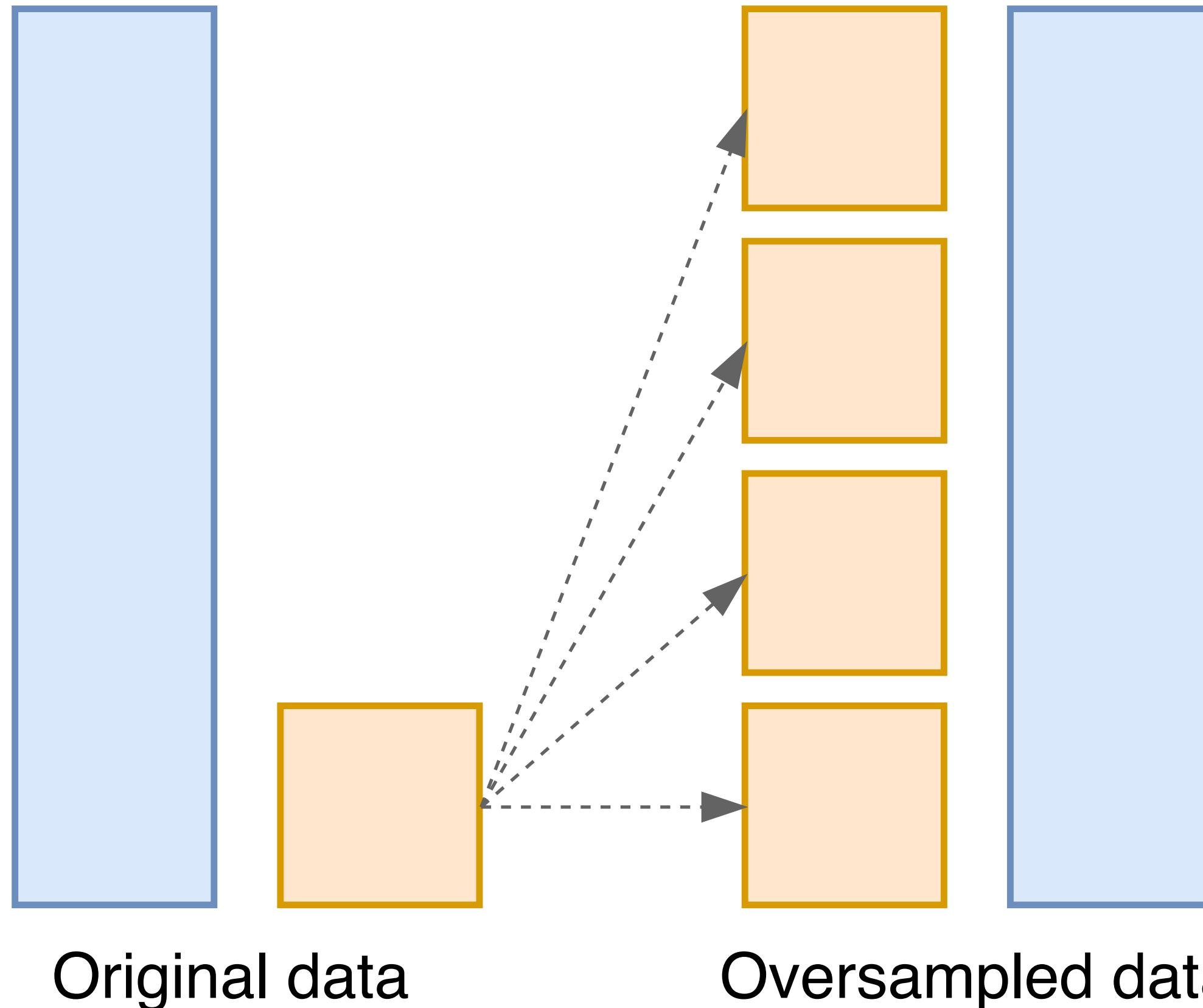
Exemple de techniques pour le texte

# Données déséquilibrées

- Distribution inégale des étiquettes dans les données étiquetées
  - ▶ Ex 2 classes: 90% Pos, 10% Neg, Ratio 4:1
- Techniques
  - ▶ Changer de métriques
    - matrice de confusion, précision/rappel, F1
    - Kappa, Courbe ROC
  - ▶ Re-échantillonage (voir suite)
  - ▶ Génération d'exemples (SMOTE)



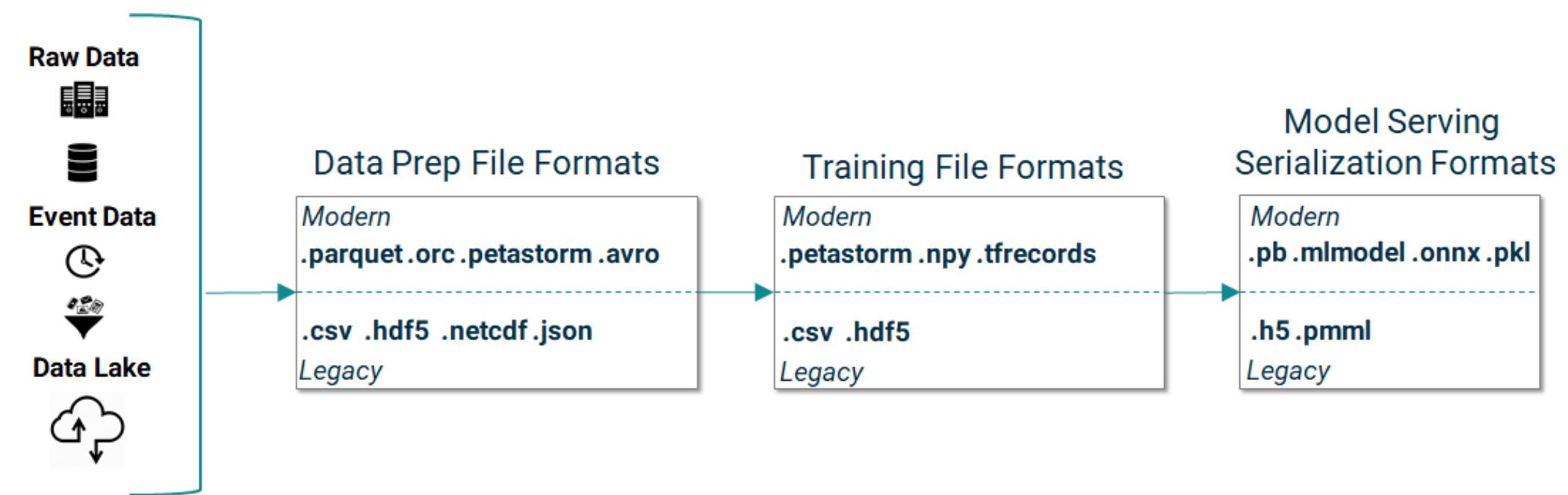
# Sur-échantillonnage vs sous-échantillonnage



Copie d'exemples de la classe minoritaire  
Création de nouveaux exemples de la classe minoritaire

Suppression d'exemples de la classe majoritaire

# Stockage des données<sup>1</sup>



- Gestion des formats d'entrée
  - ▶ colonnes, lignes
  - ▶ hiérarchique (json, hdf5)
  - ▶ vecteur, text, image, ....
- Gestion des volumes
  - ▶ file-storage, data-storage, object-storage

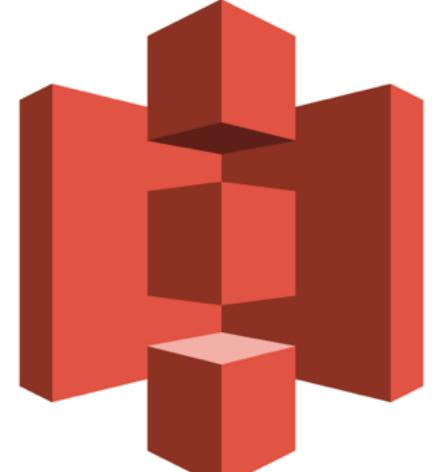
## Row Format

1	john	4.1
2	mike	3.5
3	sally	6.4



## Column Format

1	2	3
john	mike	sally
4.1	3.5	6.4



Amazon S3

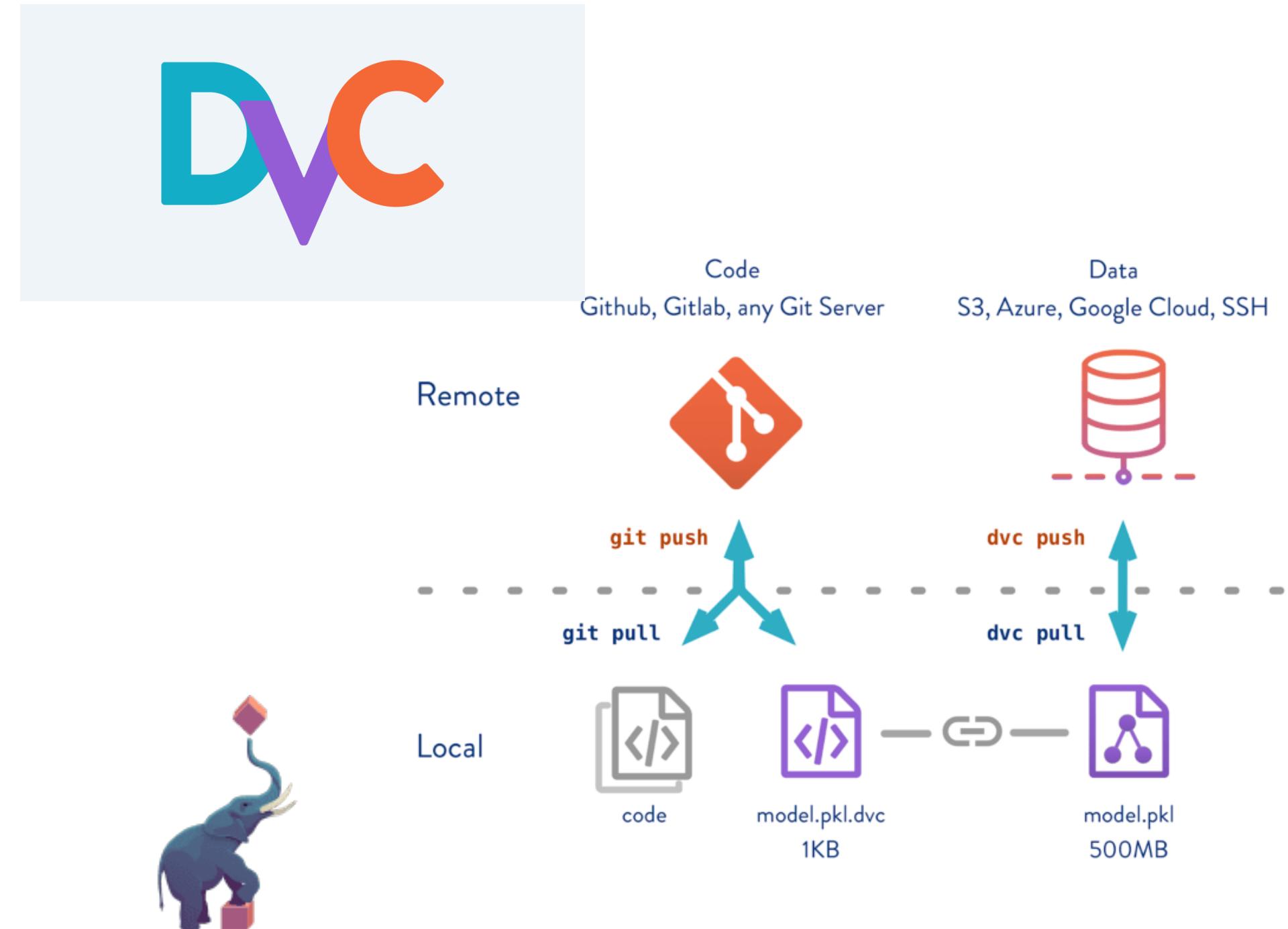


Google  
BigQuery



# Stockage des données<sup>2</sup>

- Gestion des versions
  - Maj des données => Maj des caractéristiques => Maj des modèles
  - Etiquetages collaboratifs, reproductibilité
  - Retour arrière en cas de dégradation performance
- Documentation et métadonnées
  - Signification des données
  - Comment elles ont été collectées, ou les méthodes utilisées pour les créer
  - Détails des étapes de prétraitement et des fractionnements train-validation-test
  - Explication de toutes les données qui ont été exclues
  - Format utilisé pour stocker les données, types d'attributs ou de caractéristiques



**Pachyderm**  
version-controlled data science

The screenshot shows the Pachyderm web interface with several tabs open in a browser:

- Pachdash - Google Chrome (active tab)
- Pachdash
- Stats by gabrielg
- Github Marketplace
- Issues - pachyderm
- dash/datum-card
- Job state shows
- ember-split-string
- ember.js - How do I...
- ember.js - How do I...
- JavaScript - JS review
- Other bookmarks

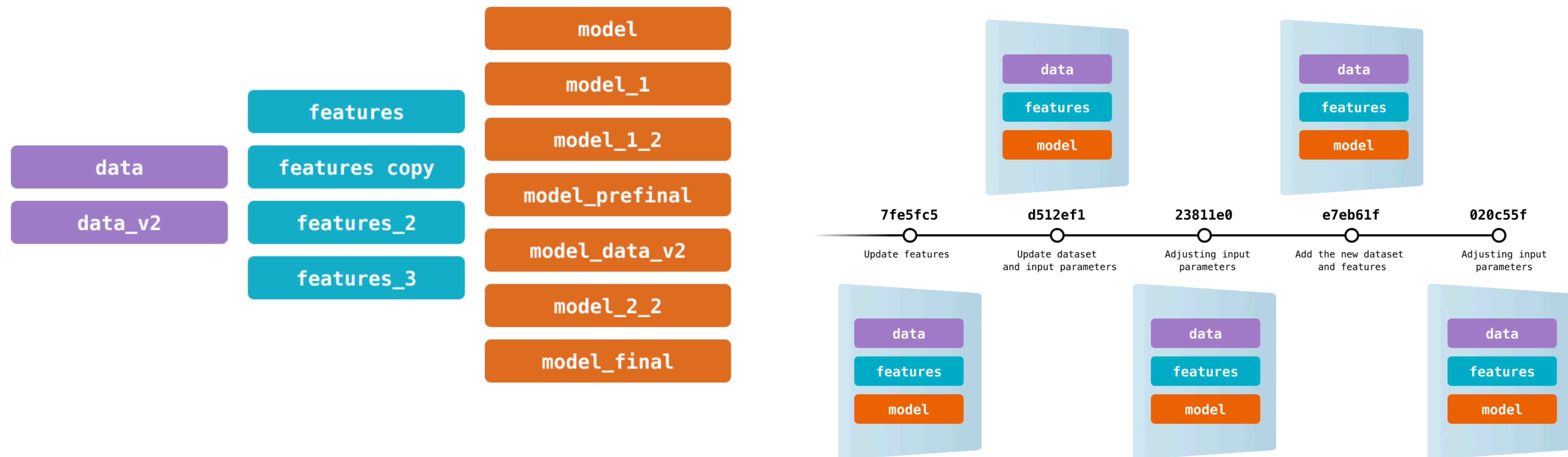
The main Pachdash view displays a failed job for pipelinea9174d927932/7abd2374. Key metrics shown include:

- Created by Pipeline: pipelinea9174d927932
- Runtime: 356ms total runtime
- Data Transfer: 22.7ms download time, 4.73ms processing time, 328ms upload time
- Input/Output: 400 datums total, 145 datums processed, 0 B input data, 0 B output data

Below the main view, there are sections for Transform Details (using ubuntu:16.04, bash command, stdln input, and accepted return codes 0) and Logs from (processed by worker). The logs section shows:

- Logs: 18 lines • 3.55 KB • Download
- 1 process call started - request: job\_id:"7abd2374-4341-4d4b-84ef-2a3f7fb8524c" data:<file\_info:<file:<commit
- 2 Received request
- 3 input has not been processed, downloading data

# Stockage des données<sup>3</sup>



# **Ingénierie des caractéristiques**

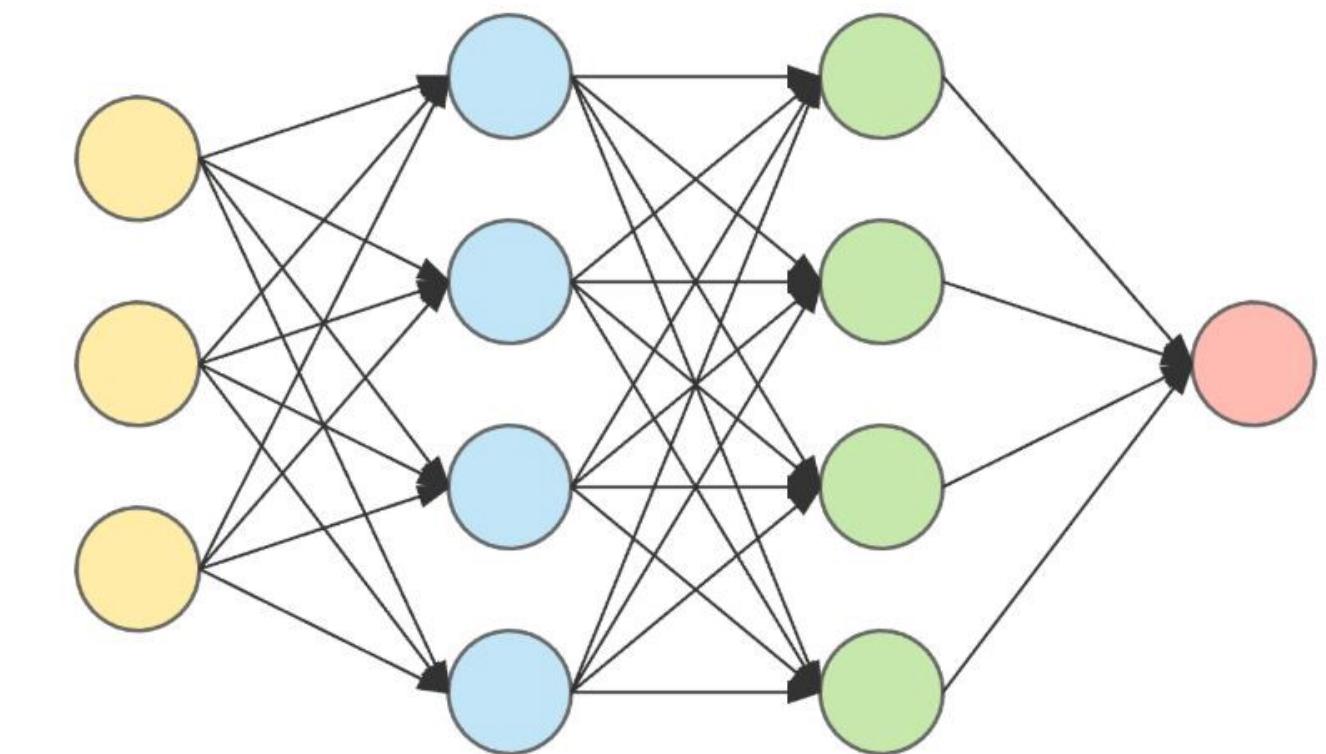
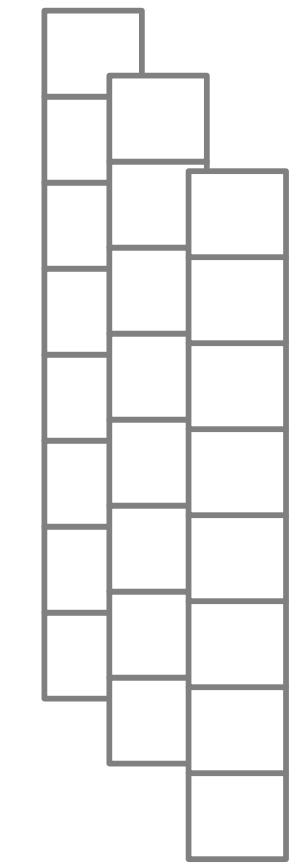
# Ingénierie des caractéristiques

- Les algorithmes d'apprentissage (classique et profond) prennent en entrées des vecteurs, matrices, tenseurs (binaire, entier, réel, ...)
- Nécessité de transformer les données brutes en une représentation acceptable par les algorithmes
- L'ingénierie des caractéristiques s'intéresse à cette transformation de façon conceptuelle et programmatique
  - Conceptualisation des caractéristiques utiles à l'apprentissage (de façon directe ou indirecte)
  - Sélection, Aggrégation, Synthèse, Apprentissage des caractéristiques
  - Schéma de transformation des exemples bruts
  - Stockage des caractéristiques
- Nécessite de la créativité, du savoir-faire, de l'expérience mais l'apprentissage profond apporte de nouvelles solutions

# Exemple du Texte

Date collected	Plot	Species	Sex	Weight
1/9/78	1	DM	M	40
1/9/78	1	DM	F	36
1/9/78	1	DS	F	135
1/20/78	1	DM	F	39
1/20/78	2	DM	M	43
1/20/78	2	DS	F	144
3/13/78	2	DM	F	51
3/13/78	2	DM	F	44
3/13/78	2	DS	F	146

transformation



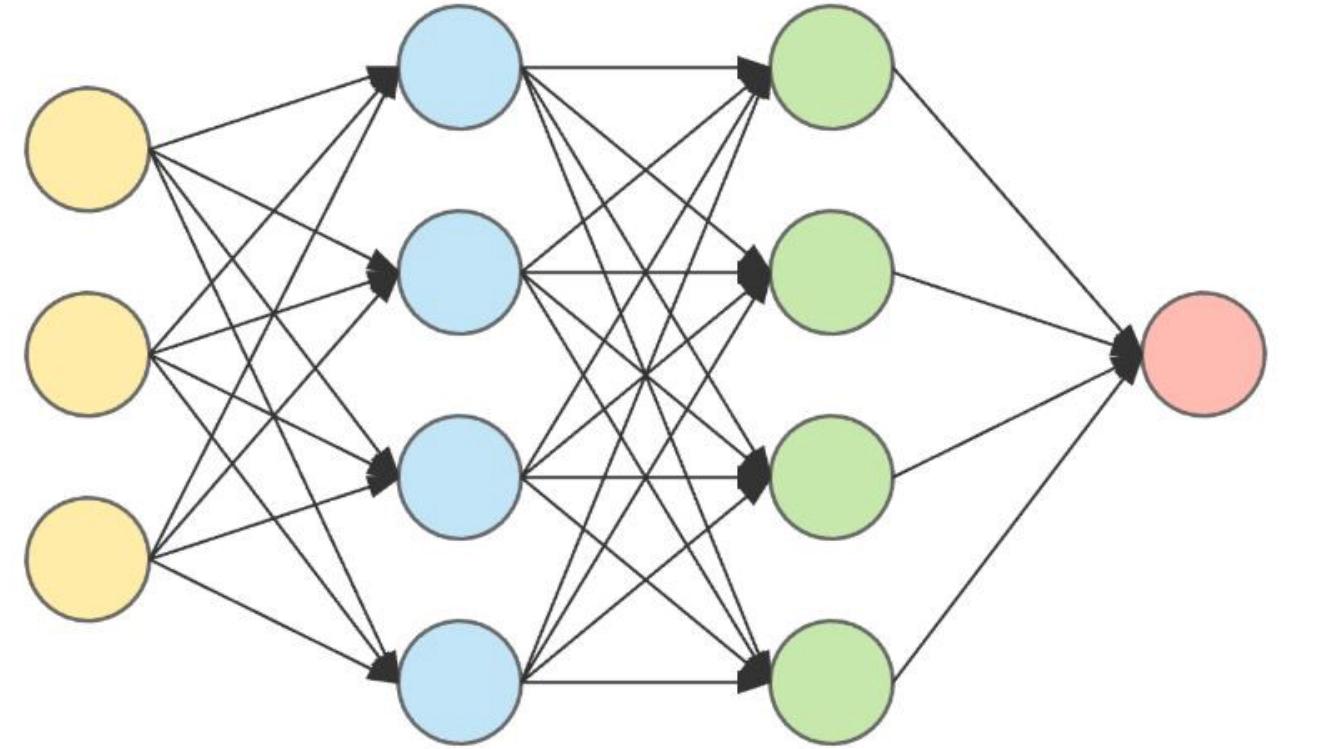
UNITED STATES

Obama tried to give Zuckerberg a wake-up call over fake news on Facebook

The Washington Post · By Adam Entous, Elizabeth...  
Economy September 25, 2017 at 7:45 AM Facebook CEO Mark Zuckerberg's company recently said it would turn over to Congress more than 3,000 politically themed

transformation

?



Apprentissage

# Caractéristiques du Texte

- Caractéristiques observées

  - ▶ Mots

    - taille, préfix, suffixe, capital

    - n-grams de caractères

    - ...

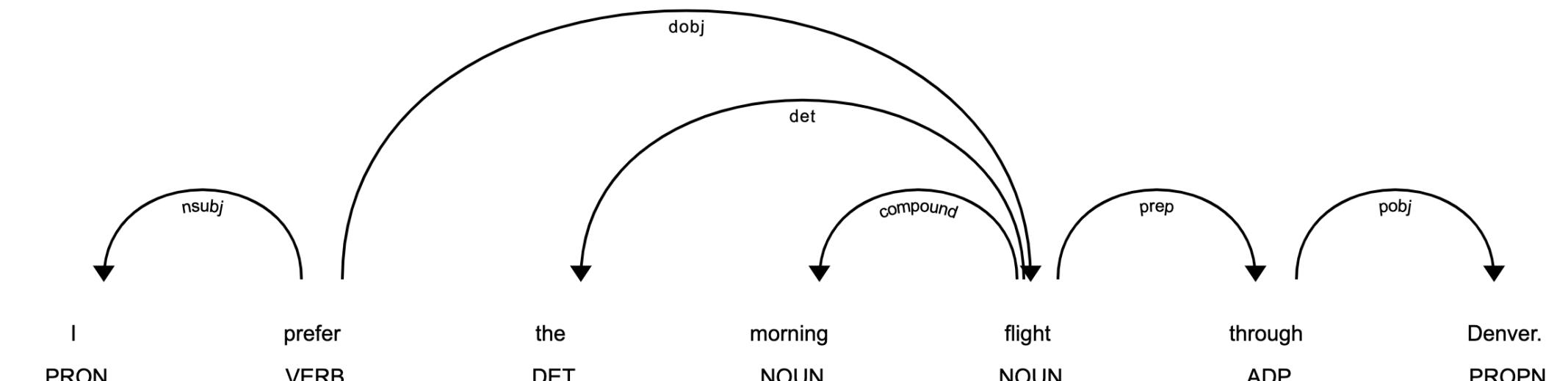
- ▶ Ensemble de mots

  - présence, comptage et poids des mots

  - séquence des mots: n-grams, colocalisation

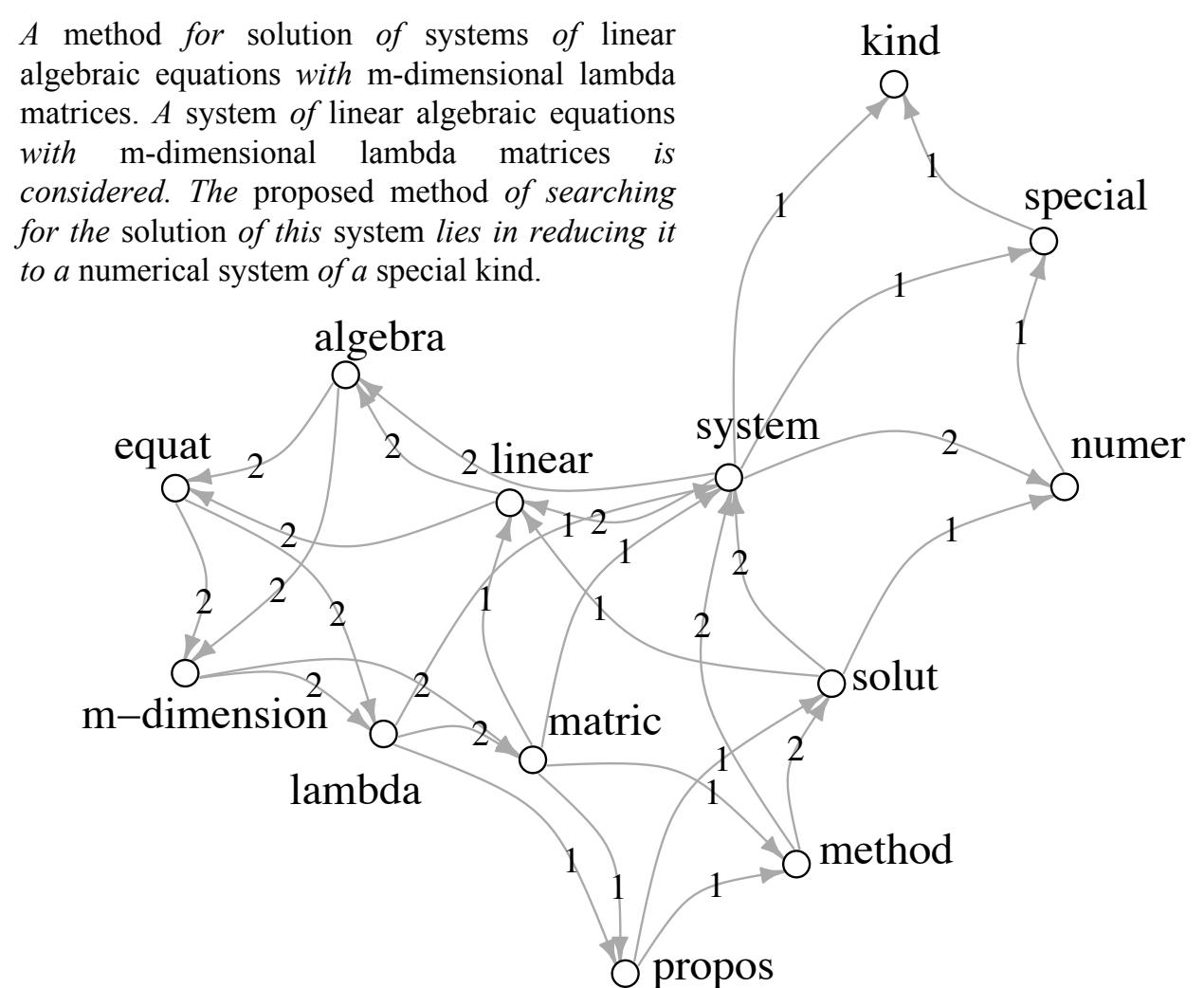
  - ...

- Caractéristiques inférées



TEXT	LEMMA	POS	TAG	DEP
Apple	apple	PROPN	NNP	nssubj
is	be	VERB	VZB	aux
looking	look	VERB	VBG	ROOT
at	at	ADP	IN	prep
buying	buy	VERB	VBG	pcomp
U.K.	u.k.	PROPN	NNP	compound
startup	startup	NOUN	NN	dobj
for	for	ADP	IN	prep
\$	\$	SYM	\$	quantmod
1	1	NUM	CD	compound
billion	billion	NUM	CD	pobj

A method for solution of systems of linear algebraic equations with m-dimensional lambda matrices. A system of linear algebraic equations with m-dimensional lambda matrices is considered. The proposed method of searching for the solution of this system lies in reducing it to a numerical system of a special kind.



# Sac de mots - Encodage One-Hot

Document 1	Love, love is a verb
Document 2	Love is a doing word
Document 3	Feathers on my breath
Document 4	Gentle impulsion
Document 5	Shakes me, makes me lighter
Document 6	Feathers on my breath



Document 1	[Love, love, is, a, verb]
Document 2	[Love, is, a, doing, word]
Document 3	[Feathers, on, my, breath]
Document 4	[Gentle, impulsion]
Document 5	[Shakes, me, makes, me, lighter]
Document 6	[Feathers, on, my, breath]

a                      breath                      doing                      feathers  
 gentle                  impulsioп                  is                      lighter  
 love                    makes                      me                      my  
 on                        shakes                     verb                    word

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Document 1	1	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0
Document 2	1	0	1	0	0	0	1	0	1	0	0	0	0	0	0	1
Document 3	0	1	0	1	0	0	0	0	0	0	0	1	1	0	0	0
Document 4	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
Document 5	0	0	0	0	0	0	0	1	0	1	1	0	0	1	0	0
Document 6	0	1	0	1	0	0	0	0	0	0	0	1	1	0	0	0



a	breath	doing	feathers	gentle	impulsion	is	lighter	love	makes	me	my	on	shakes	verb	word
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

# Conversion des étiquettes en nombres

- Concerne les attributs catégoriels
- Méthodes
  - Recherche et remplacement
    - $\text{four} \Rightarrow 4, \text{six} \Rightarrow 6, \dots$
  - Encodage d'étiquettes
    - Association d'un entier à chaque étiquette et remplacement par l'entier
  - Création d'une colonne binaire par étiquette (ex: colonne `drive_wheels`)

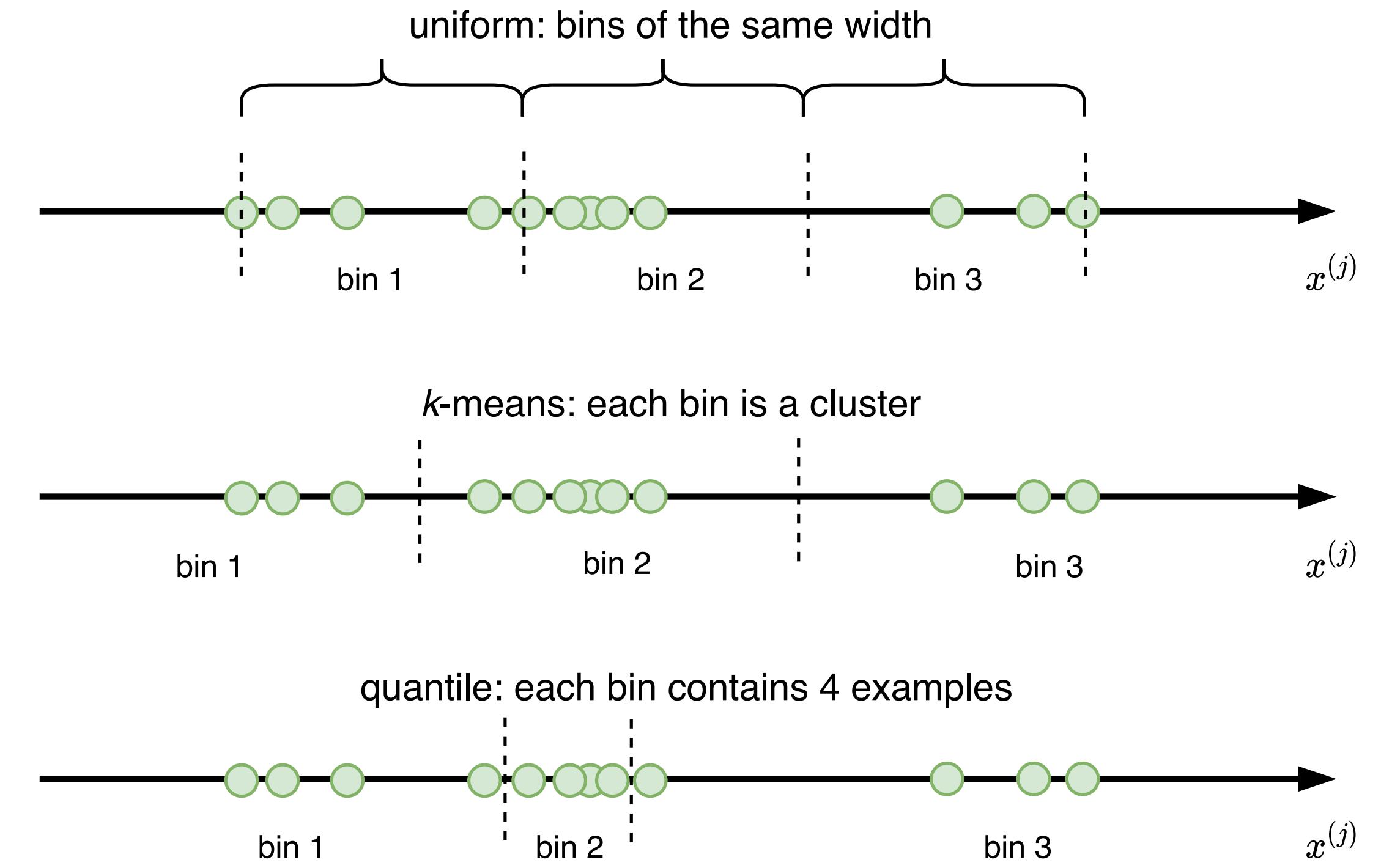
	make	fuel_type	aspiration	num_doors	body_style	drive_wheels	engine_location	engine_type	num_cylinders	fu
0	alfa-romero	gas	std	two	convertible	rwd	front	dohc	four	m
1	alfa-romero	gas	std	two	convertible	rwd	front	dohc	four	m
2	alfa-romero	gas	std	two	hatchback	rwd	front	ohcv	six	m
3	audi	gas	std	four	sedan	fwd	front	ohc	four	m
4	audi	gas	std	four	sedan	4wd	front	ohc	five	m

	make	fuel_type	aspiration	num_doors	body_style	drive_wheels	engine_location	engine_type	num_cylinders	fu
0	alfa-romero	gas	std	2	convertible	rwd	front	dohc	4	m
1	alfa-romero	gas	std	2	convertible	rwd	front	dohc	4	m
2	alfa-romero	gas	std	2	hatchback	rwd	front	ohcv	6	m
3	audi	gas	std	4	sedan	fwd	front	ohc	4	m
4	audi	gas	std	4	sedan	4wd	front	ohc	5	m

style_cat	body_convertible	body_hardtop	body_hatchback	body_sedan	body_wagon	drive_4wd	drive_fwd	drive_rwd
	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0
	1.0	0.0	0.0	0.0	0.0	0.0	0.0	1.0

# Discrétisation de valeurs numériques

- Regroupement en groupes de valeurs discrets
  - Groupes d'âge
  - Groupe selon des tranches de salaire
- Peut conduire à une meilleure précision de prédiction
- Ajoute des informations utiles à l'algorithme d'apprentissage lorsque l'ensemble de données d'apprentissage est relativement petit.
- Peut-être plus simple pour un humain d'interpréter la prédiction
- Techniques de Binning



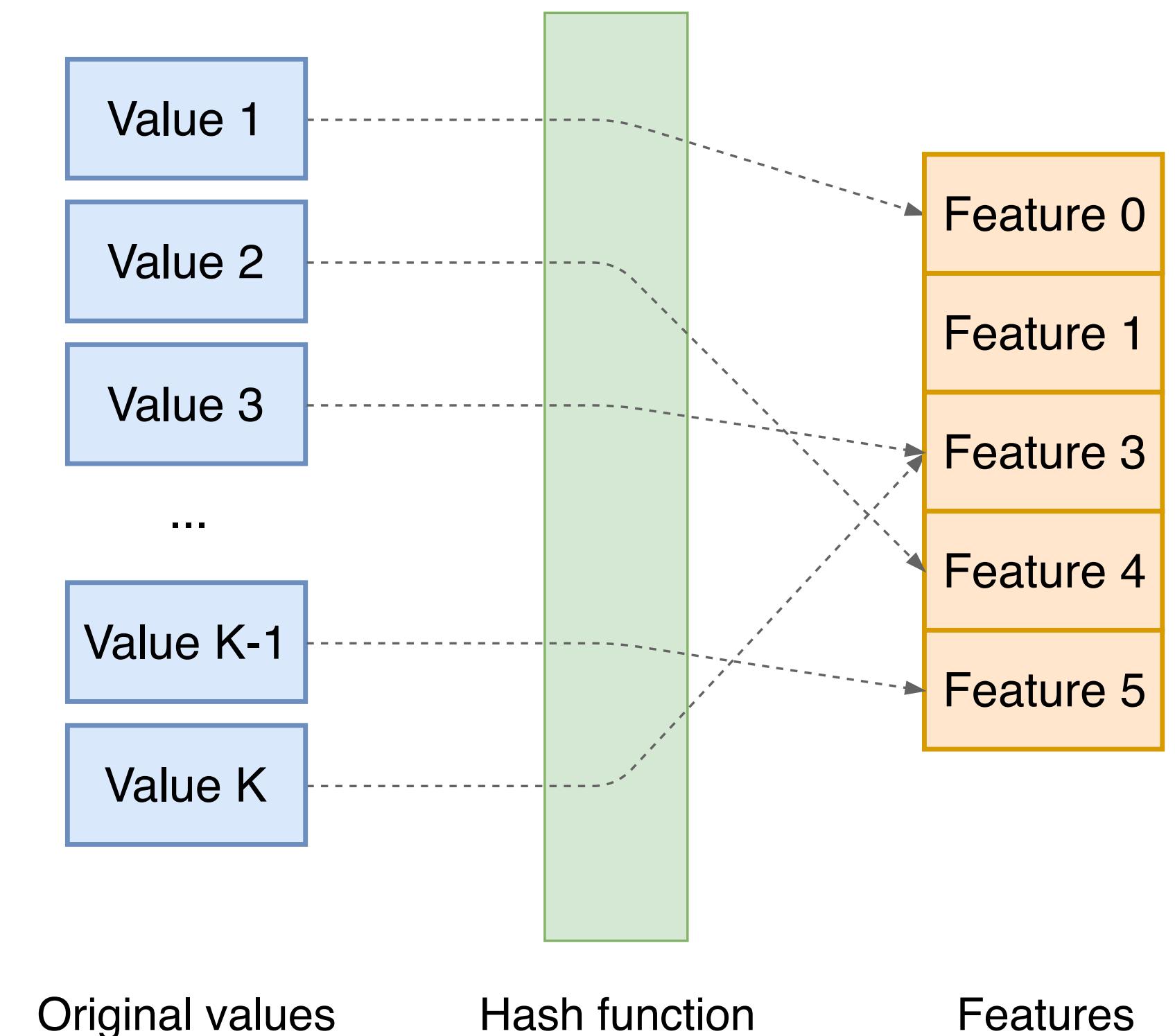
# Hachage de caractéristiques

- Utile pour les données textuelles ou les attributs catégoriels avec de nombreuses valeurs
- Objectif: limiter le nombre de dimensions
- Principe:
  - ▶ Conversion de toutes les valeurs de l'attribut catégoriel (ou tous les tokens de la collection de documents) en un nombre avec une fonction de hachage
  - ▶ Conversion du nombre en un index du vecteur de caractéristiques

$h(\text{love}) \bmod 5 = 0$   
 $h(\text{is}) \bmod 5 = 3$   
 $h(\text{a}) \bmod 5 = 1$   
 $h(\text{doing}) \bmod 5 = 3$   
 $h(\text{word}) \bmod 5 = 4.$

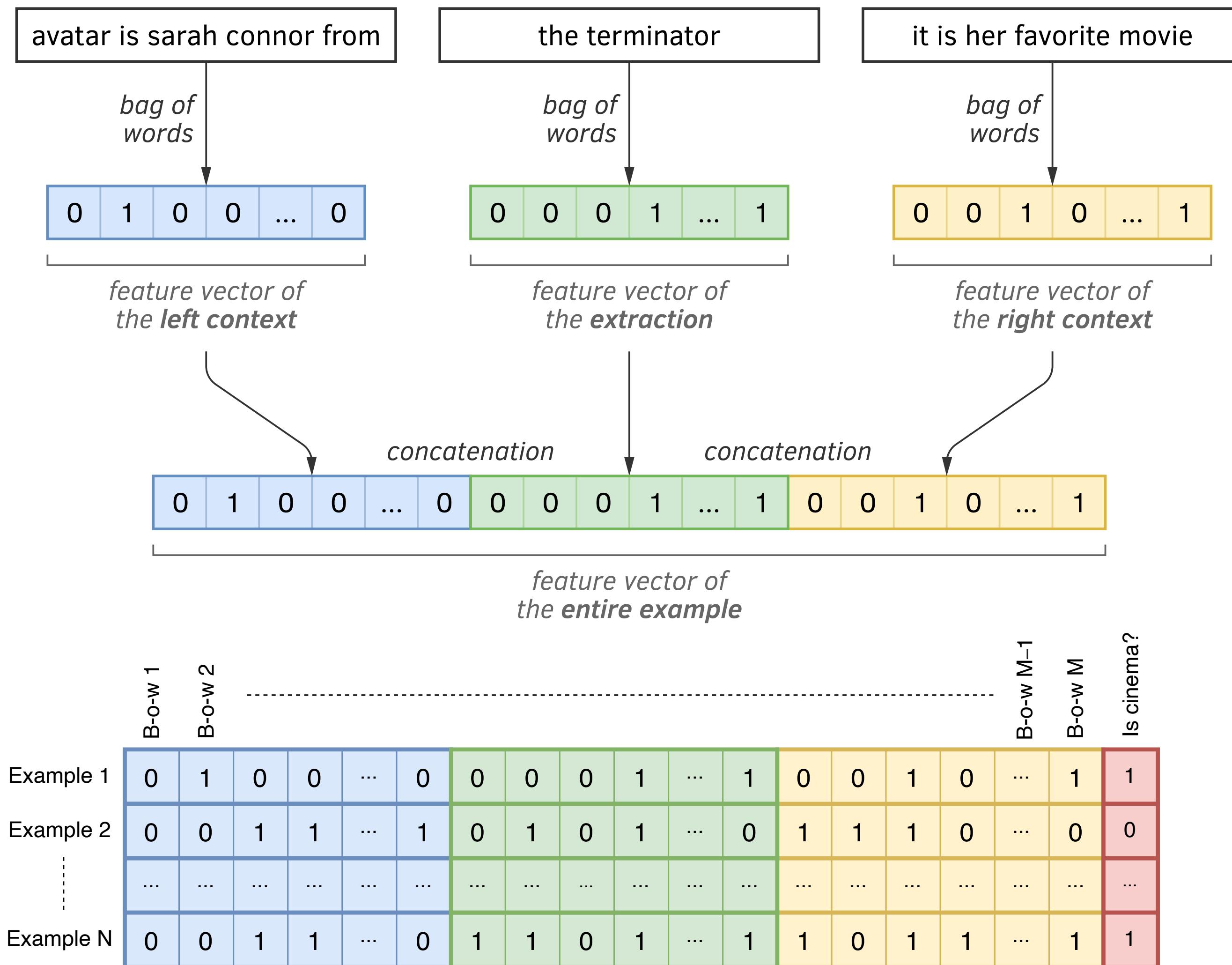
“Love is a doing word”  
[1, 1, 0, 2, 1]

a	breath	doing	feathers	gentle	impulsion	is	lighter	love	makes	me	my	on	shakes	verb	word
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16



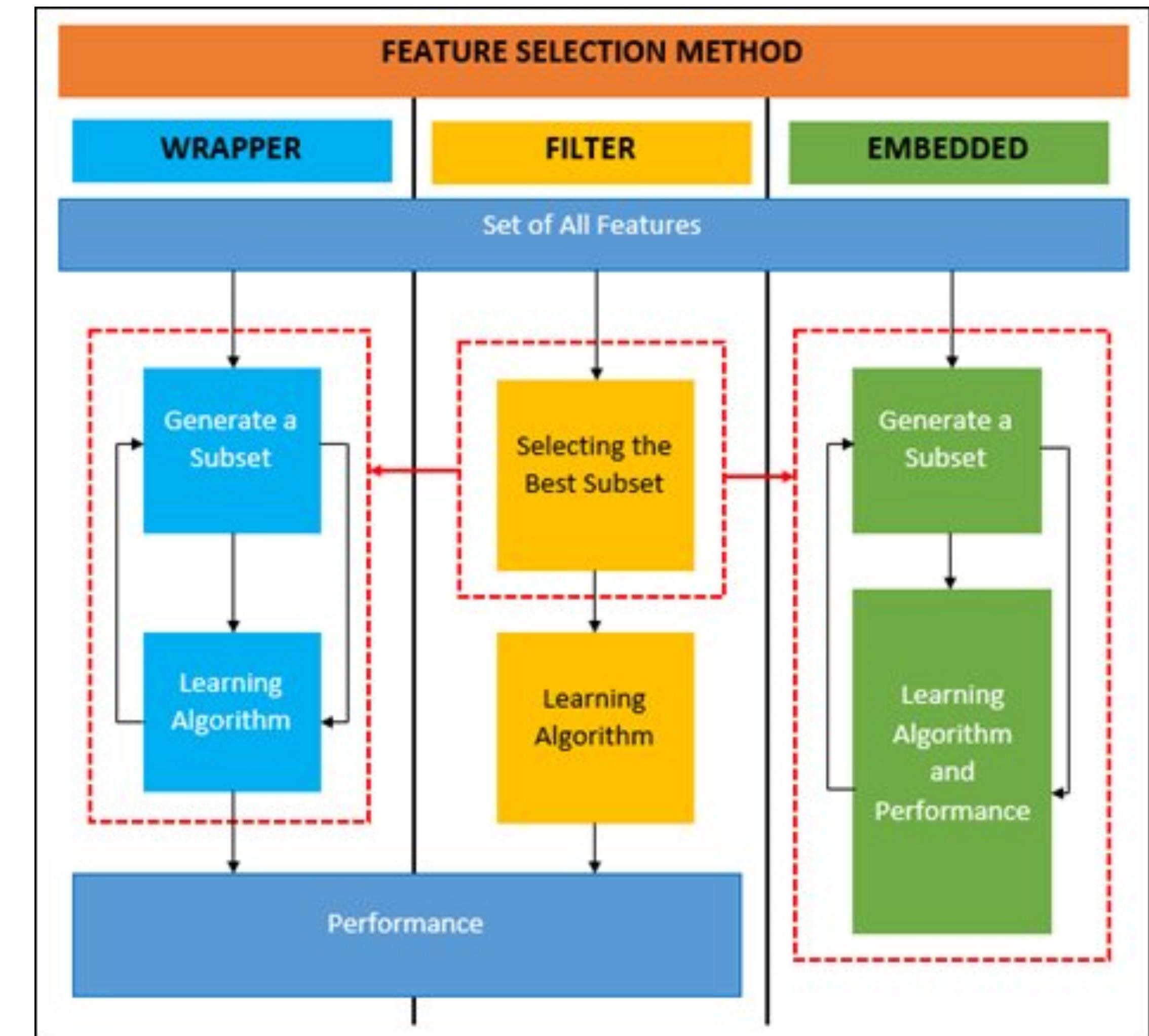
# Empilement de caractéristiques

- Combinaison de plusieurs textes
  - ▶ ex: requête / réponse, texte + sujet
- Combinaison de données et du texte
  - ▶ ex: attributs d'un produit accompagné de l'avis utilisateur
- L'ordre n'est pas important en général
- Respect du même ordre pour tous les exemples



# Réduction des caractéristiques

- Objectifs
  - ▶ Eviter un vecteur de caractéristiques trop large qui peut conduire à des temps d'apprentissage prohibitif
  - ▶ Diminuer la taille globale des données pour tenir dans la mémoire vive
- Plusieurs méthodes
  - ▶ Eliminer les caractéristiques redondantes (mesure de corrélation)
  - ▶ Aggréger plusieurs caractéristiques en une seule
  - ▶ Méthodes trouvant les caractéristiques les plus importantes:
    - Evaluation de la corrélation entre chaque variable d'entrée et le résultat en sortie
    - Sélection de sous-ensemble et évaluation des résultats
  - ▶ Réduction de dimensions : PCA par exemple



# Synthèse de caractéristiques

- Aider l'algorithme d'apprentissage à apprendre en lui fournissant un ensemble plus riche de caractéristiques
- Construction
  - ▶ A partir des données relationnelles
    - Statistiques sur des colonnes de tables en lien avec la donnée
  - ▶ A partir des autres caractéristiques
    - Discrétisation (vue précédemment)
  - Elévation au carré, Moyenne à partir des plus proches voisins, opérations arithmétiques

User	User ID	Gender	Age	...	Date Subscribed
1	M	18	...	2016-01-12	
2	F	25	...	2017-08-23	
3	F	28	...	2019-12-19	
4	M	19	...	2019-12-18	
5	F	21	...	2016-11-30	

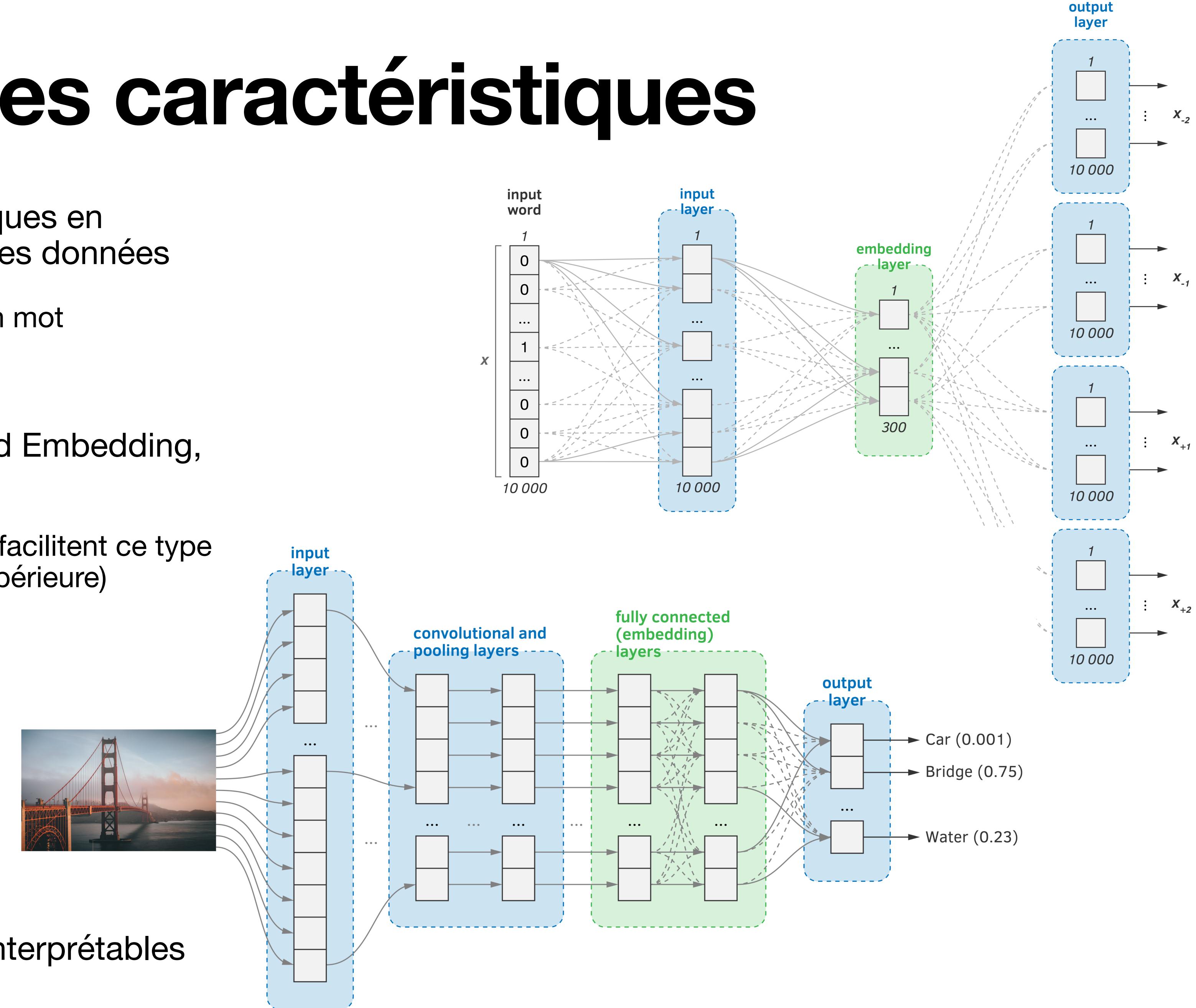
Order	Order ID	User ID	Amount	...	Order Date
1	2	23	...	2017-09-13	
2	4	18	...	2018-11-23	
3	2	7.5	...	2019-12-19	
4	2	8.3	...	2016-11-30	

Call	Call ID	User ID	Call Duration	...	Call Date
1	4	55	...	2016-01-12	
2	2	235	...	2016-01-13	
3	3	476	...	2016-12-17	
4	4	334	...	2019-12-19	
5	4	14	...	2016-11-30	

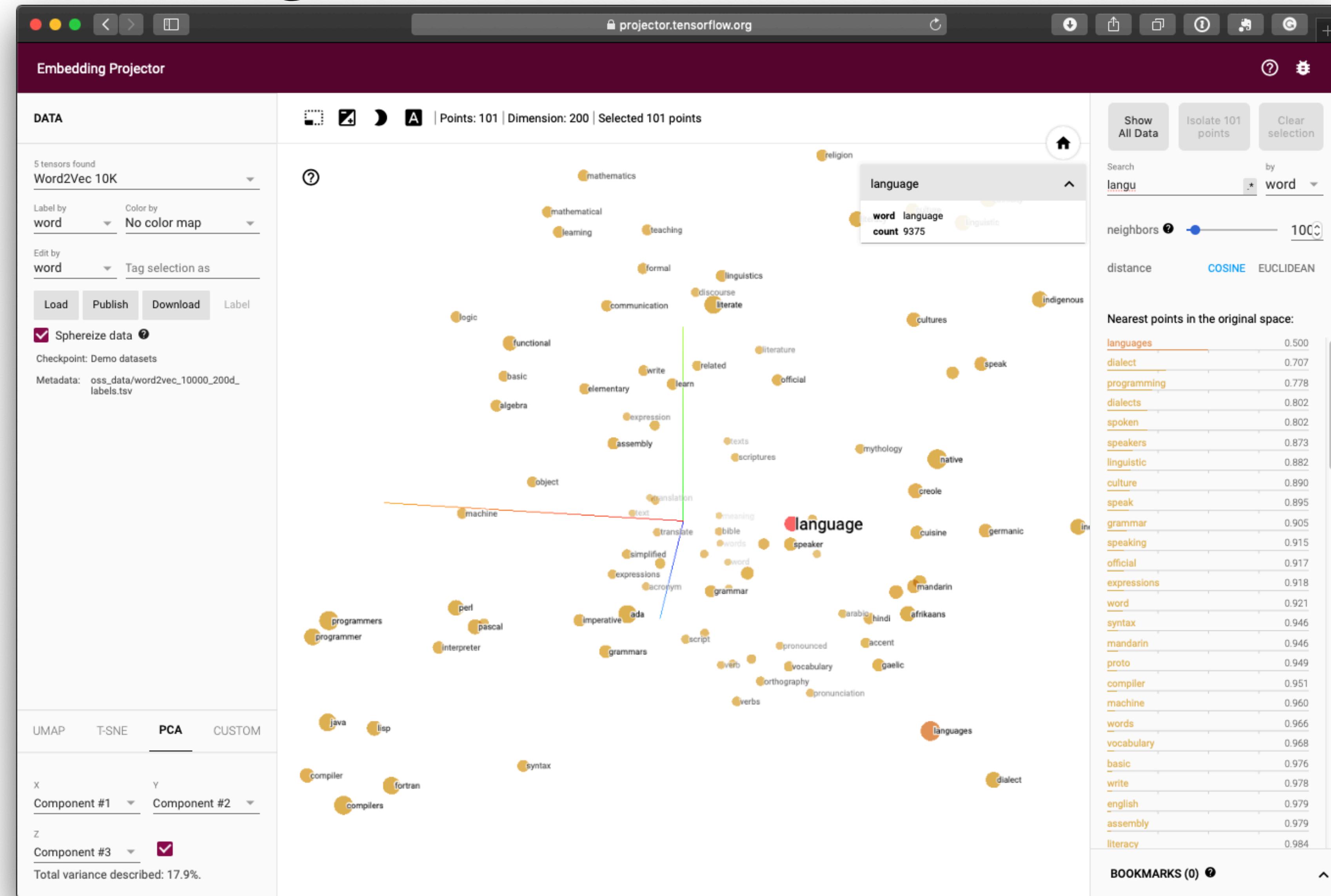
User features	User ID	Gender	Age	Mean Order Amount	Std Dev Order Amount	Mean Call Duration	Std Dev Call Duration
2	F	25	12.9	7.1	235	0	
4	M	19	18	0	134.3	142.7	

# Apprentissage des caractéristiques

- Il est possible d'apprendre des caractéristiques en exploitant les relations (non linéaires) dans les données
  - ▶ cooccurrence de mots, prédiction du prochain mot
  - ▶ motifs de région similaires dans les images
- Technique très en vogue actuellement (Word Embedding, Image Embedding, ...)
  - ▶ promu par réseaux de neurones profonds qui facilitent ce type d'apprentissage (sorties près de la couche supérieure)
  - ▶ fonctionne bien pour tout type de donnée
    - Text
    - Image
    - Son
    - ....
- Les caractéristiques apprises ne sont pas interprétables

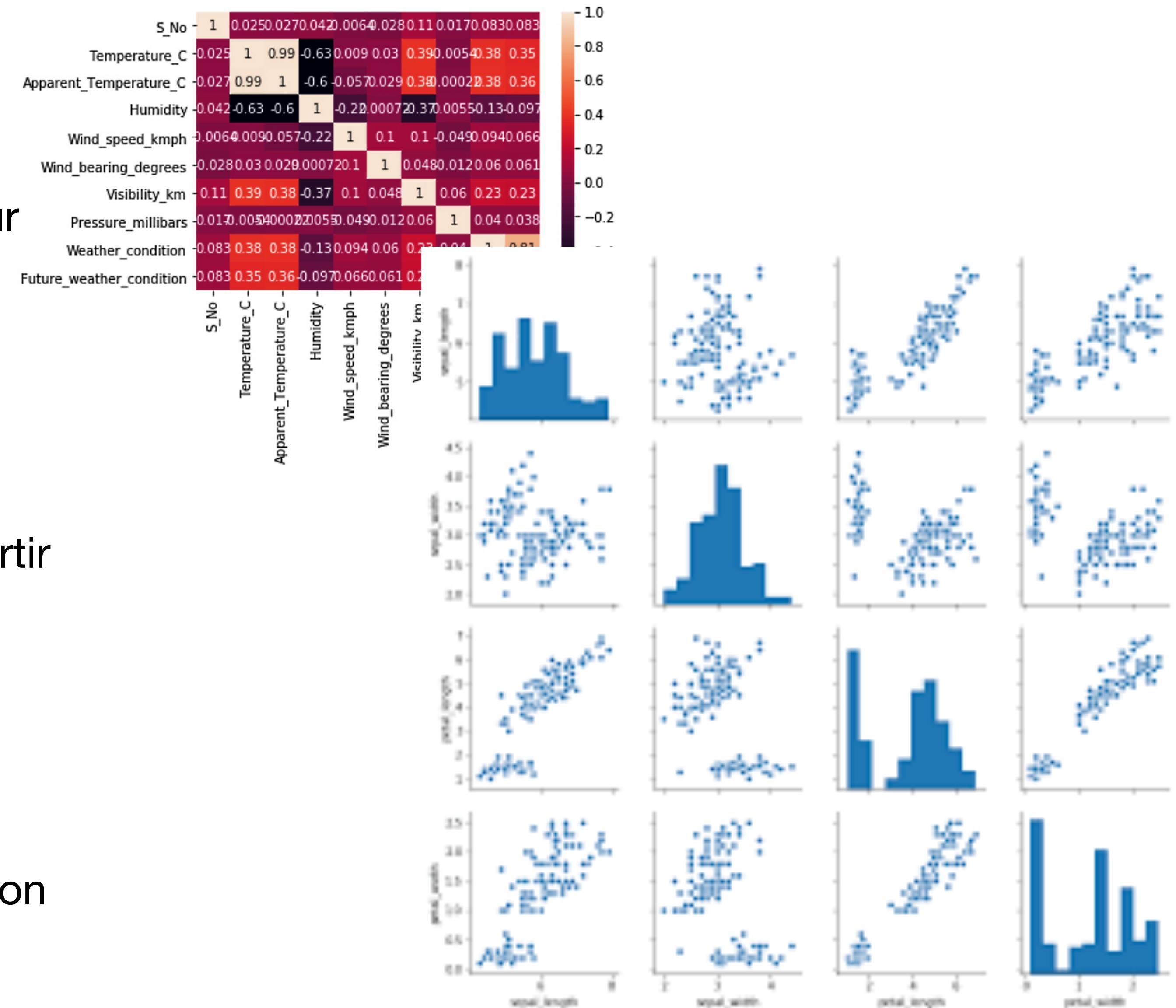


# Apprentissage de caractéristiques



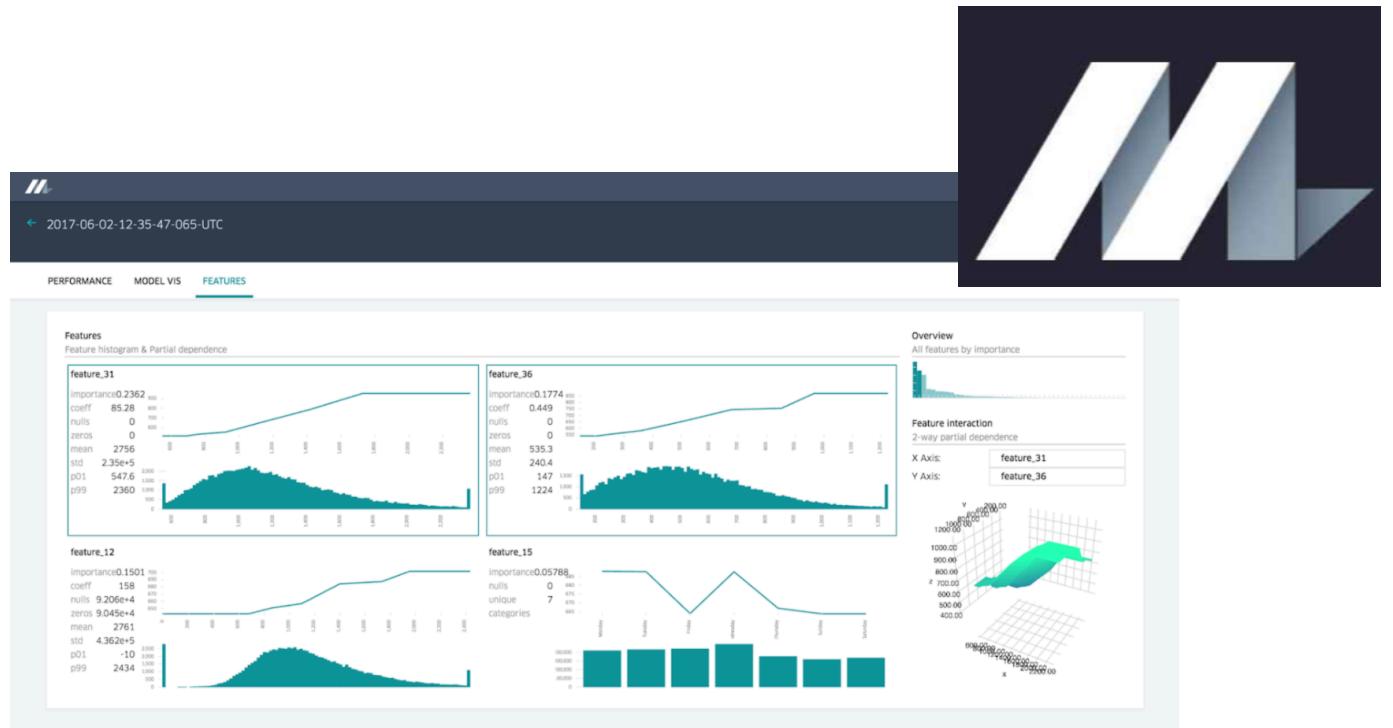
# Propriétés des bonnes caractéristiques

- Pouvoir prédictif élevé
  - ▶ Ex: Voiture + Marié ne sont pas utile pour détection du cancer
- Calcul rapide
- Fiabilité
  - ▶ Ex: Une caractéristique déterminée à partir d'un serveur est non fiable
- Non-corrélation
- Distribution des valeurs
  - ▶ Similarité entre entraînement et production

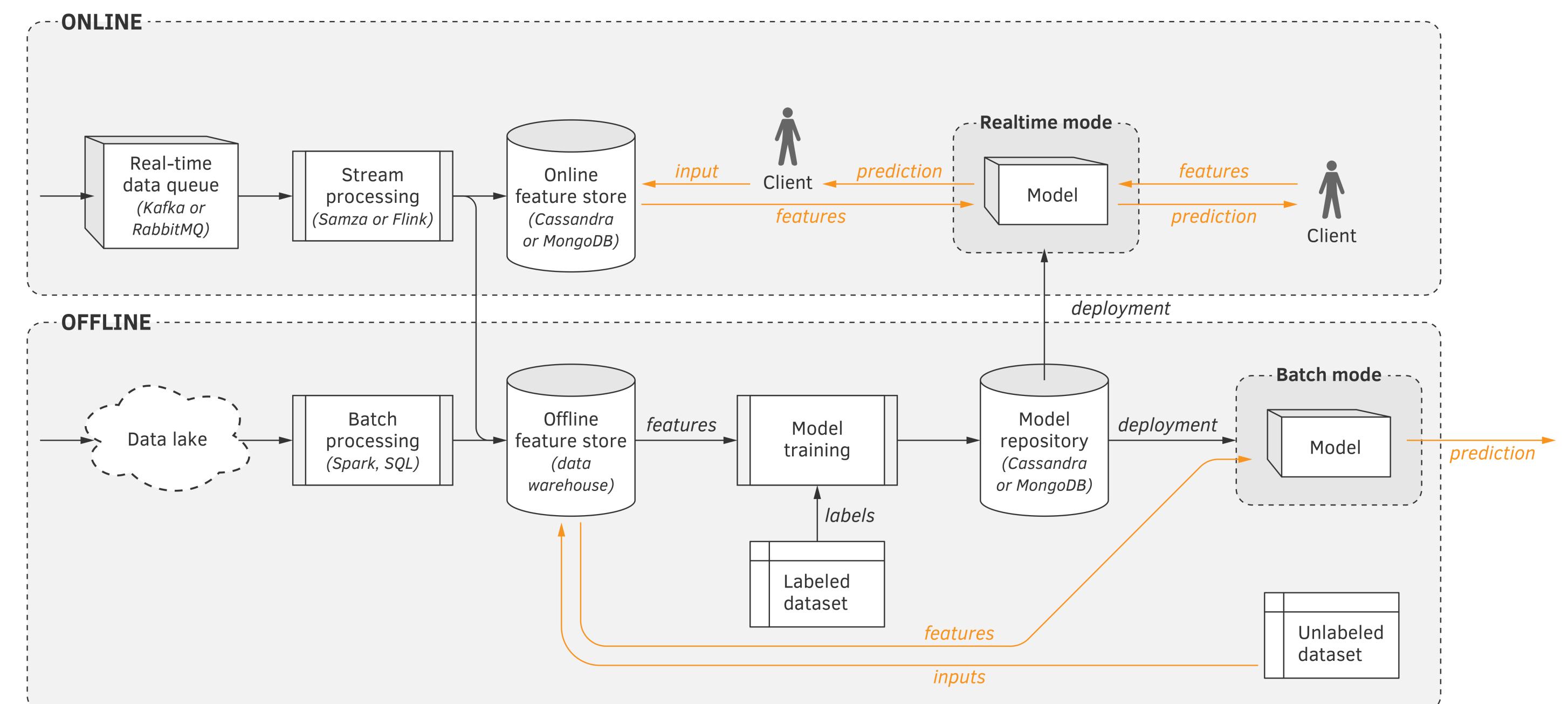


# Stockage des caractéristiques

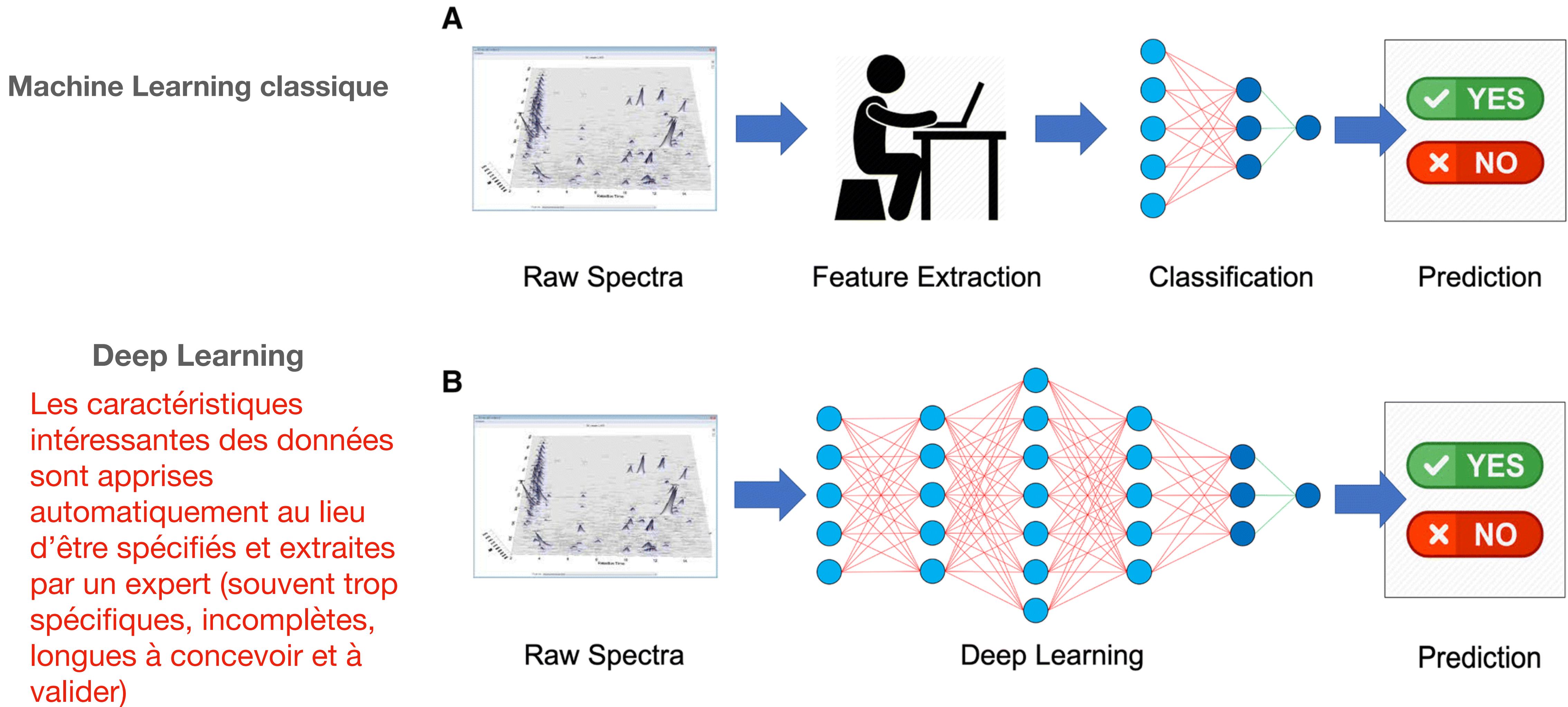
- Besoins
  - Mutualiser les caractéristiques des données dans une organisation (cohérence inter-projets, reproductibilité)
  - Éviter des traitement répétitifs pour construire, sélectionner, apprendre les caractéristiques
- Magasin de caractéristiques
  - Stockage de caractéristiques
  - Gestion des métadonnées sur les caractéristiques
  - Requête
  - Conservation des versions



```
1  feature {  
2      name : "height"  
3      type : float  
4      min : 50.0  
5      max : 300.0  
6      mean : 160.0  
7      variance : 17.0  
8      zeroes : false  
9      undefined : false  
10     popularity : 1.0  
11 }
```



# Ingénierie des caractéristiques & Deep Learning



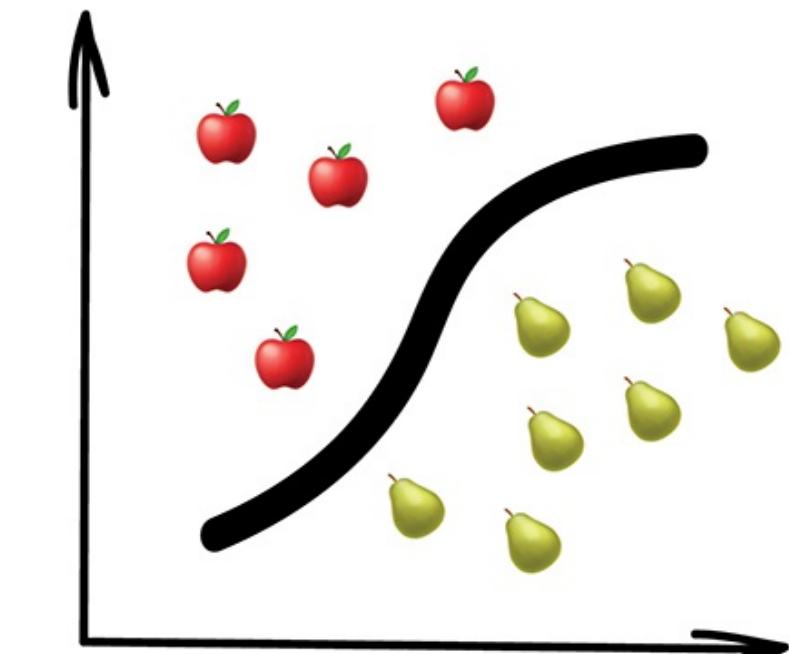
# **Entrainement et évaluation**

# Principales tâches

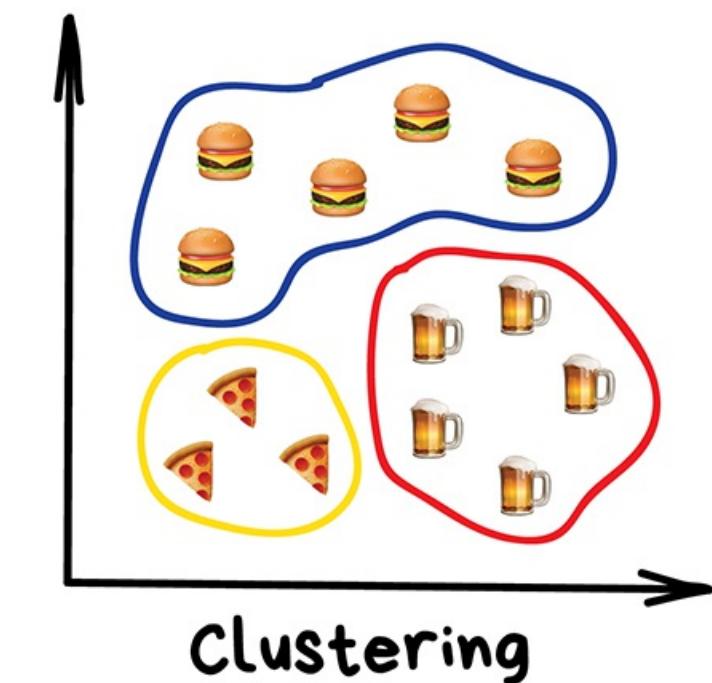
- Préparation de l'apprentissage
  - Choix de l'algorithme
  - Préparation des données d'entraînement
  - Définition du niveau de performance réalisable
- Entrainement
  - Construction du programme d'entraînement
  - Règlage des hyperparamètres
  - Evaluation des performances
- Sauvegarde du modèle

# Quelle est la nature du problème ?

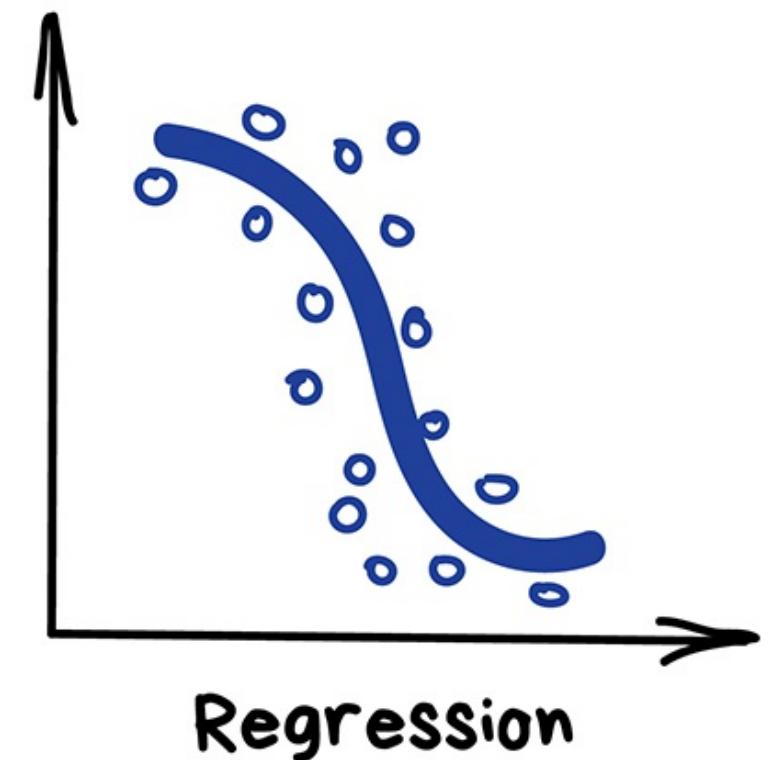
- **Classification** : attribuer une catégorie à chaque élément (par exemple, classification de documents).
- **Régression** : prédire une valeur réelle pour chaque élément (prédiction de la valeur des actions, des variables économiques).
- **Regroupement** : partitionner les données en régions "homogènes" (analyse de très grands ensembles de données).
- **Réduction de la dimensionnalité** : trouver une représentation de dimension inférieure préservant certaines propriétés des données.
- **Apprentissage de règles**: trouver des relations intéressantes entre variables dans les données



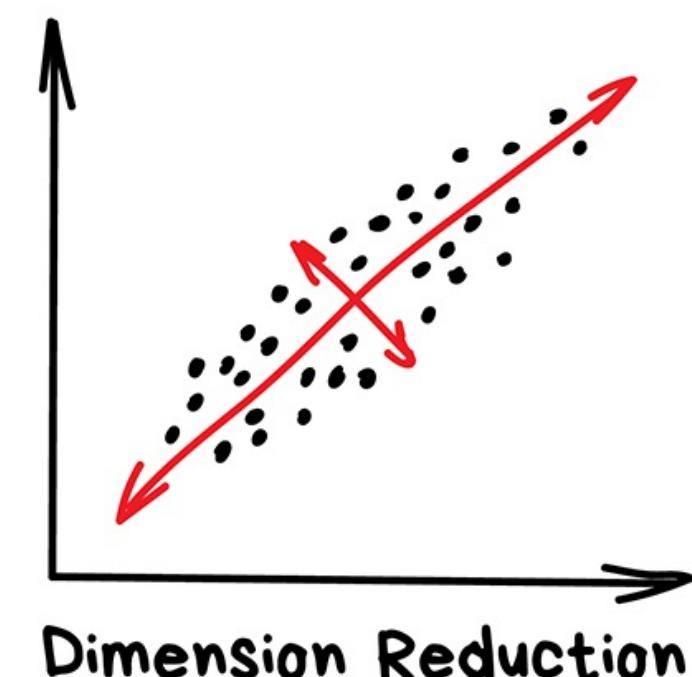
Classification



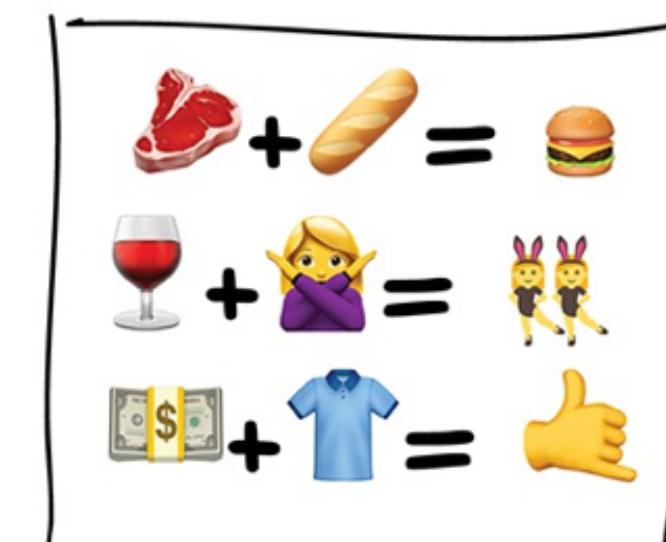
Clustering



Regression

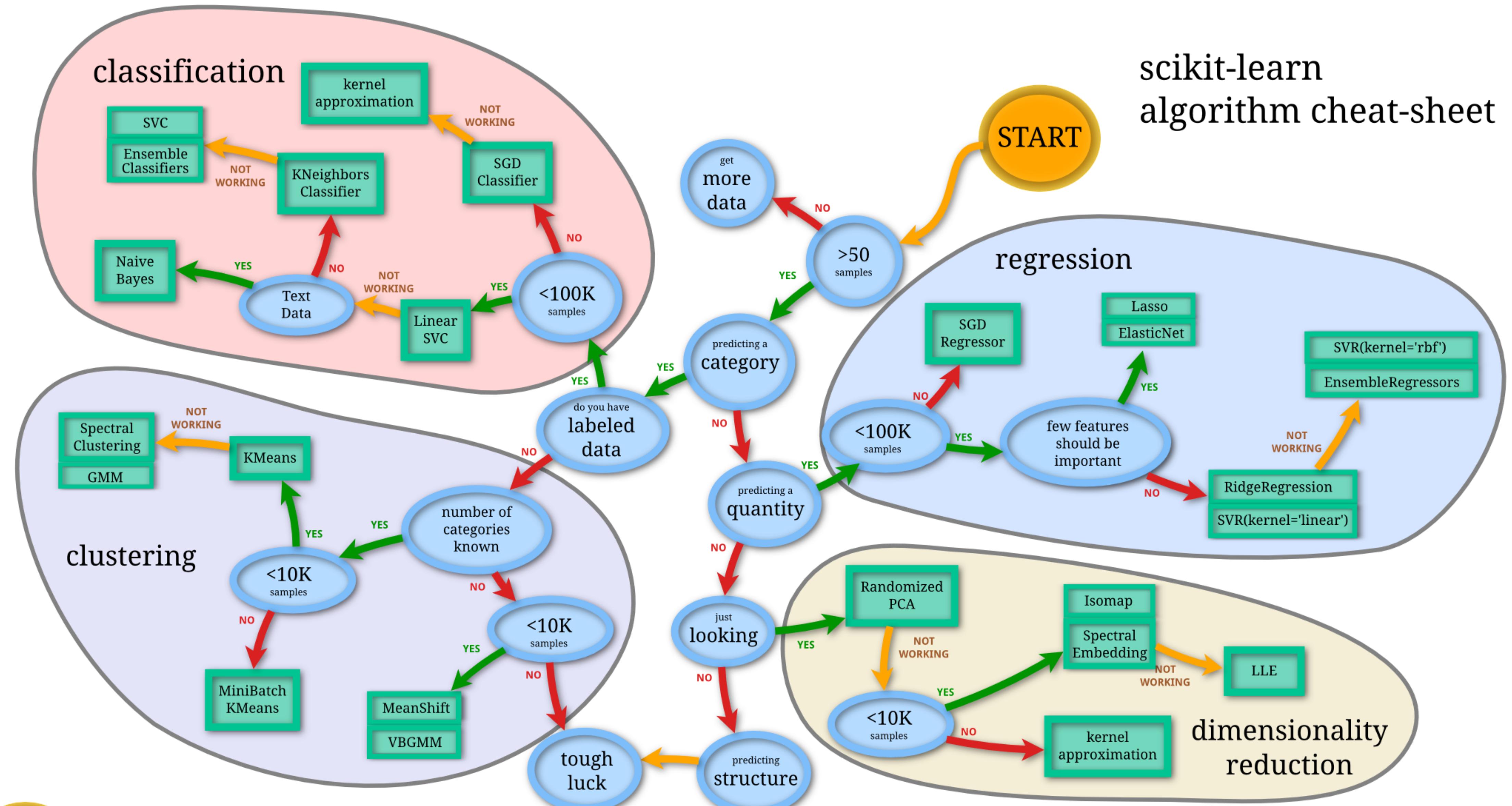


Dimension Reduction

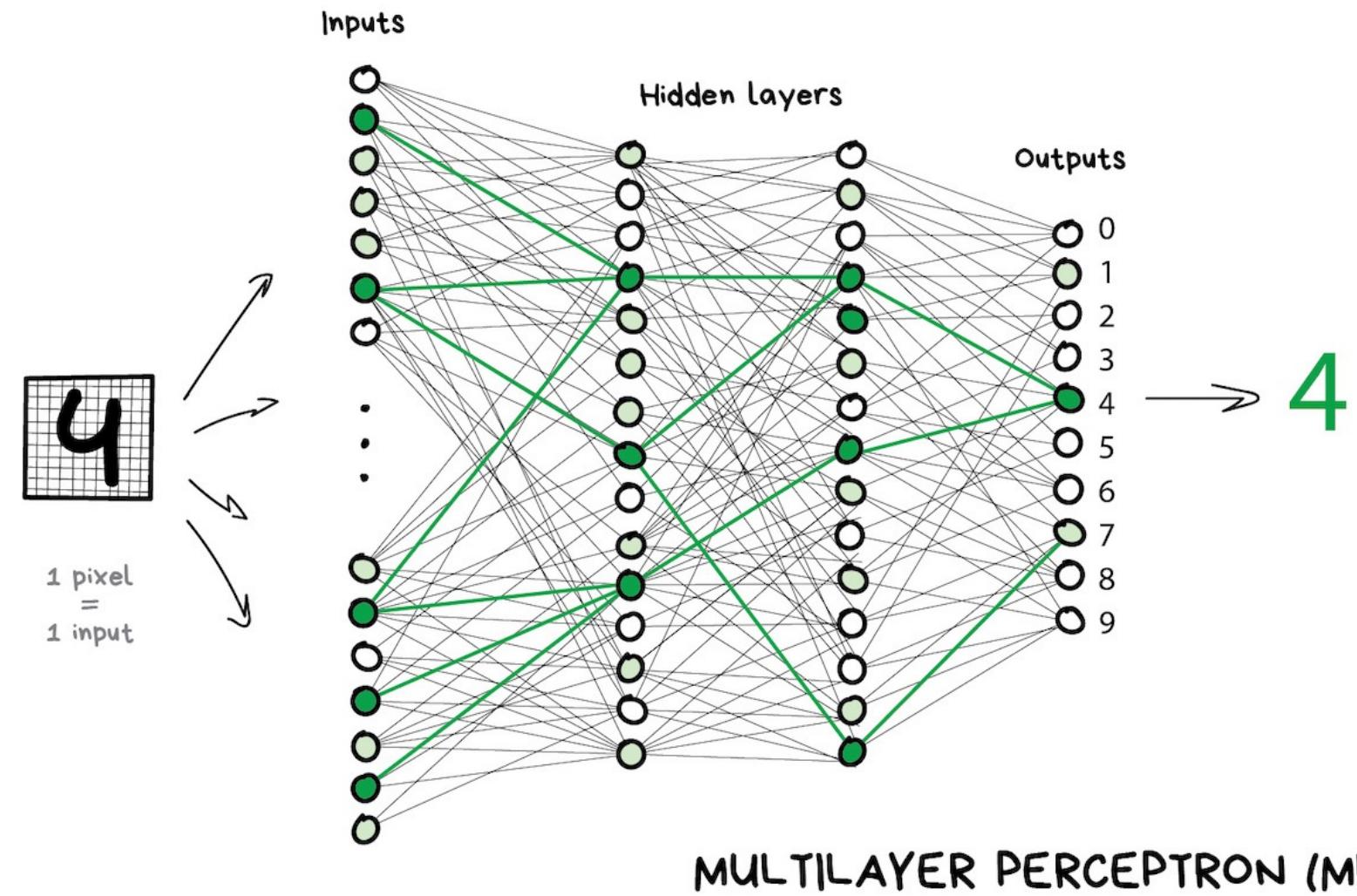


Association  
Rule Learning

# scikit-learn algorithm cheat-sheet



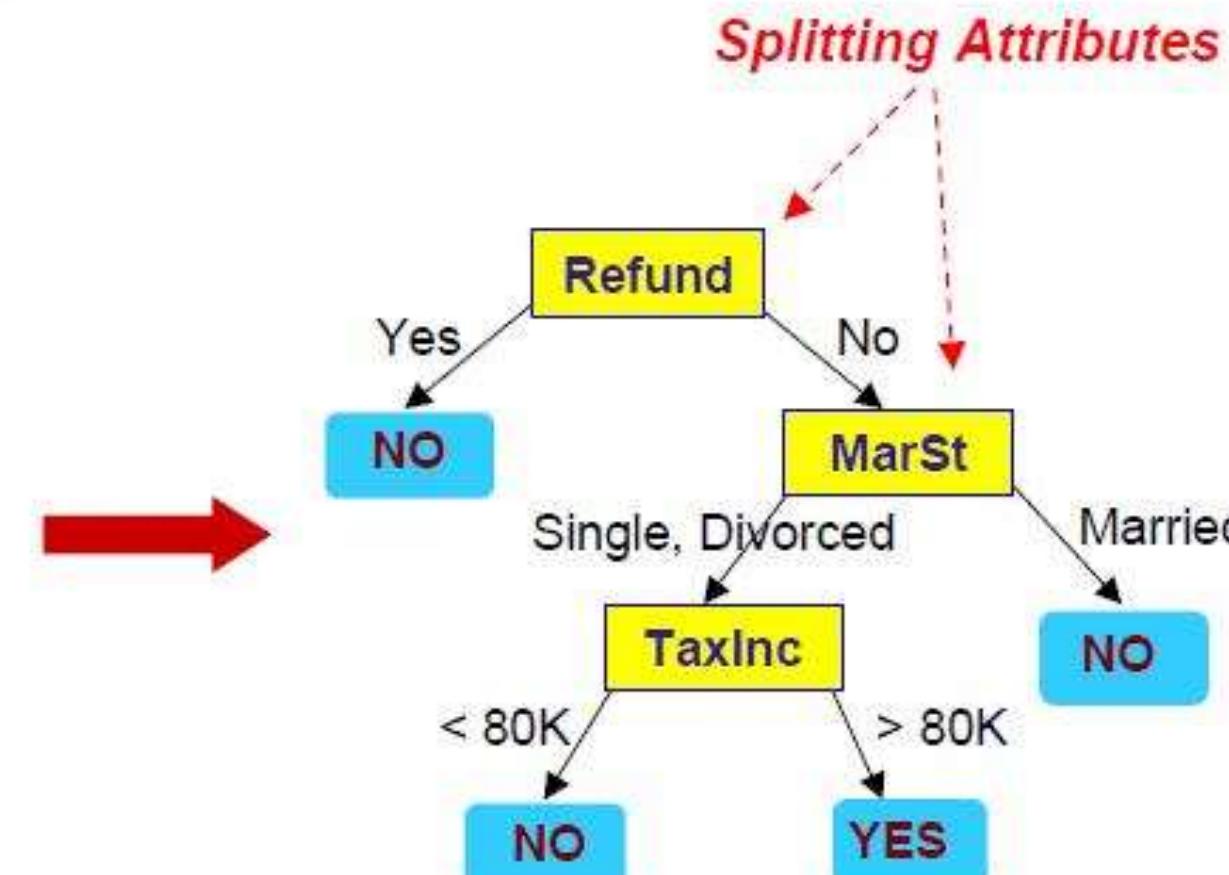
# Algorithmes pour la classification



MULTILAYER PERCEPTRON (MLP)

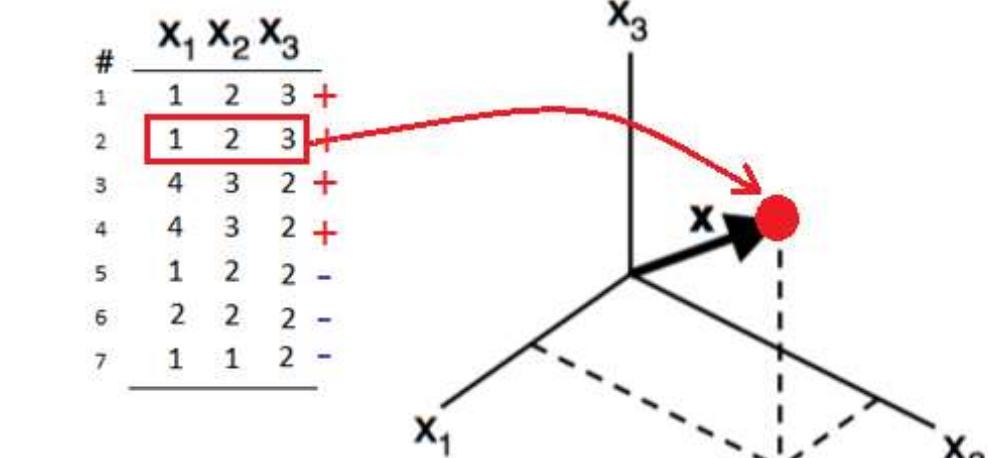
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Training Data

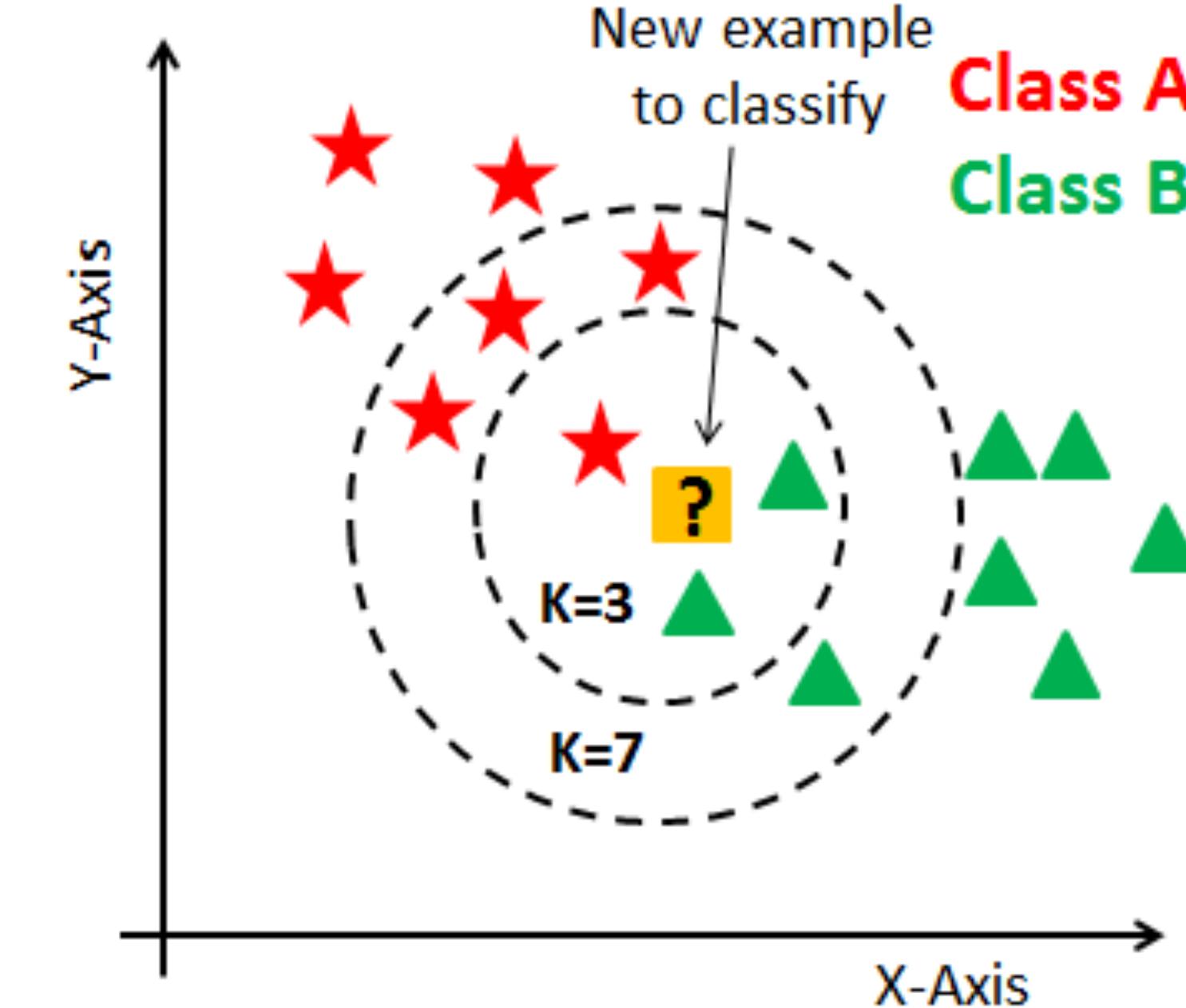


Model: Decision Tree

$$\mathbf{x} = \begin{bmatrix} \mathbf{x}_1 \\ \mathbf{x}_2 \\ \mathbf{x}_3 \end{bmatrix} \begin{array}{l} \text{Age} \\ \text{Weight} \\ \text{Pressure} \end{array}$$



Feature vector      Feature space (3D)



# Questions pour l'algorithme d'apprentissage

- **Explicabilité**
  - Les prédictions du modèle nécessitent-elles une explication ?
- **En mémoire ou hors mémoire ?**
  - L'ensemble de données peut-il être entièrement chargé dans la mémoire vive ? Si non, algorithme d'apprentissage incrémental
- **Nombre de caractéristiques et d'exemples**
  - Combien de caractéristiques chaque exemple possède-t-il ? Certains algorithmes peuvent gérer un nombre considérable d'exemples et des millions de caractéristiques (Réseaux de neurones, Ensembles)
- **Non-linéarité des données**
  - Vos données sont-elles linéairement séparables ? Peuvent-elles être modélisées à l'aide d'un modèle linéaire ?
- **Vitesse d'apprentissage**
  - Combien de temps l'algorithme d'apprentissage doit-il être exécuté pour construire un modèle et à quelle fréquence devrez-vous réentraîner le modèle sur des données actualisées ?
- **Vitesse de prédiction**
  - Quelle doit être la vitesse du modèle lorsqu'il génère des prédictions ? (penser au modèle en production)

# Vérification ponctuelle des algorithmes

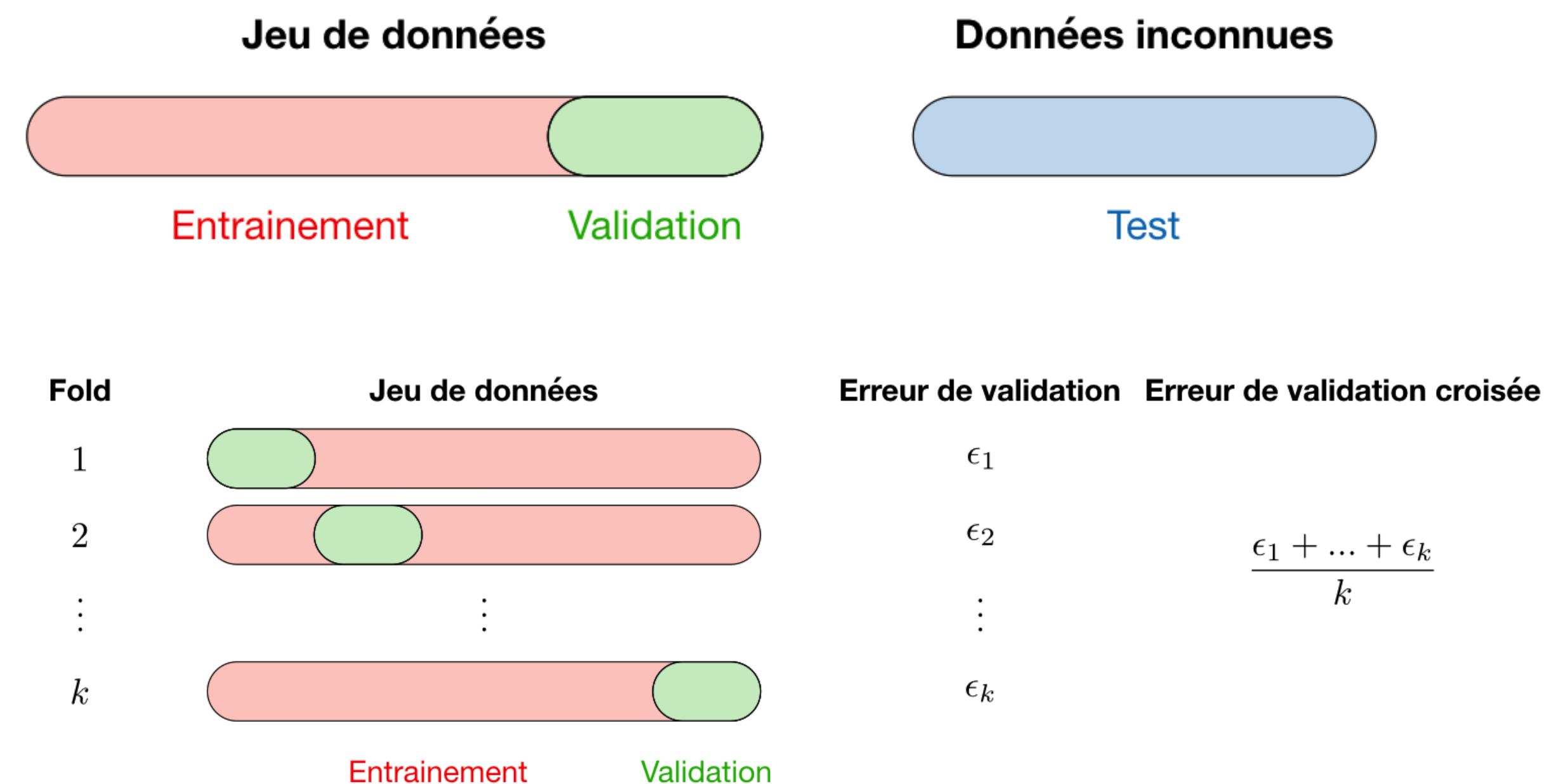
- Sélectionner des algorithmes basés sur différents principes (parfois appelés orthogonaux), tels que les algorithmes basés sur les instances, les algorithmes basés sur les noyaux, l'apprentissage superficiel, l'apprentissage profond, les ensembles ;
- Essayer chaque algorithme avec 3 à 5 valeurs différentes des hyperparamètres les plus sensibles (comme le nombre de voisins k dans les k-voisins les plus proches, la pénalité C dans les machines à vecteurs de support, ou le seuil de décision dans la régression logistique)
- Utiliser la même répartition formation/validation pour toutes les expériences
- Si l'algorithme d'apprentissage n'est pas déterministe (comme les algorithmes d'apprentissage des réseaux neuronaux et des forêts aléatoires), effectuez plusieurs expériences, puis faites la moyenne des résultats
- Une fois la vérification terminée, noter quels algorithmes ont donné les meilleurs résultats et utilisez ces informations lorsque vous travaillerez sur un problème similaire à l'avenir

# Préparation des données d'entraînement

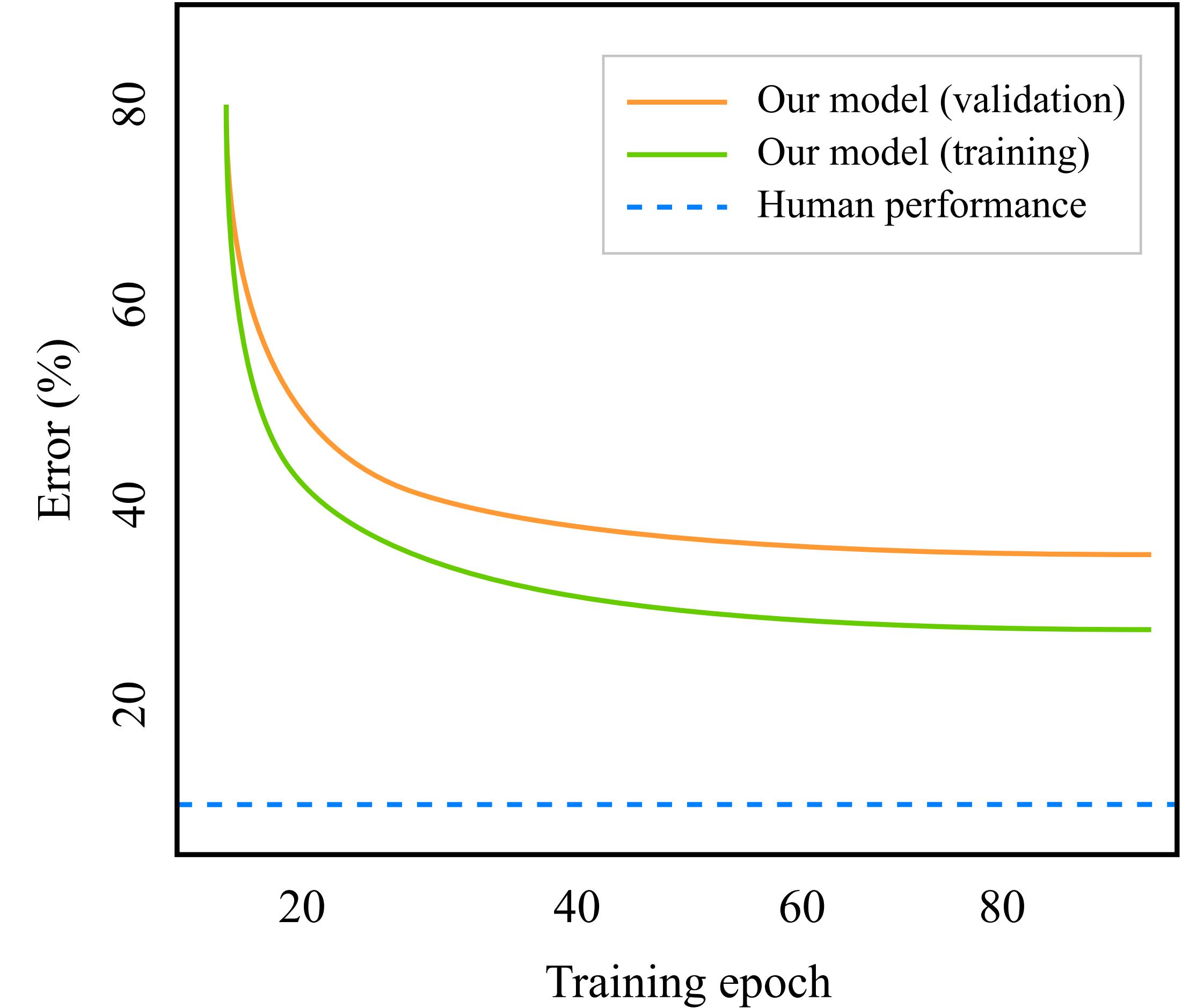
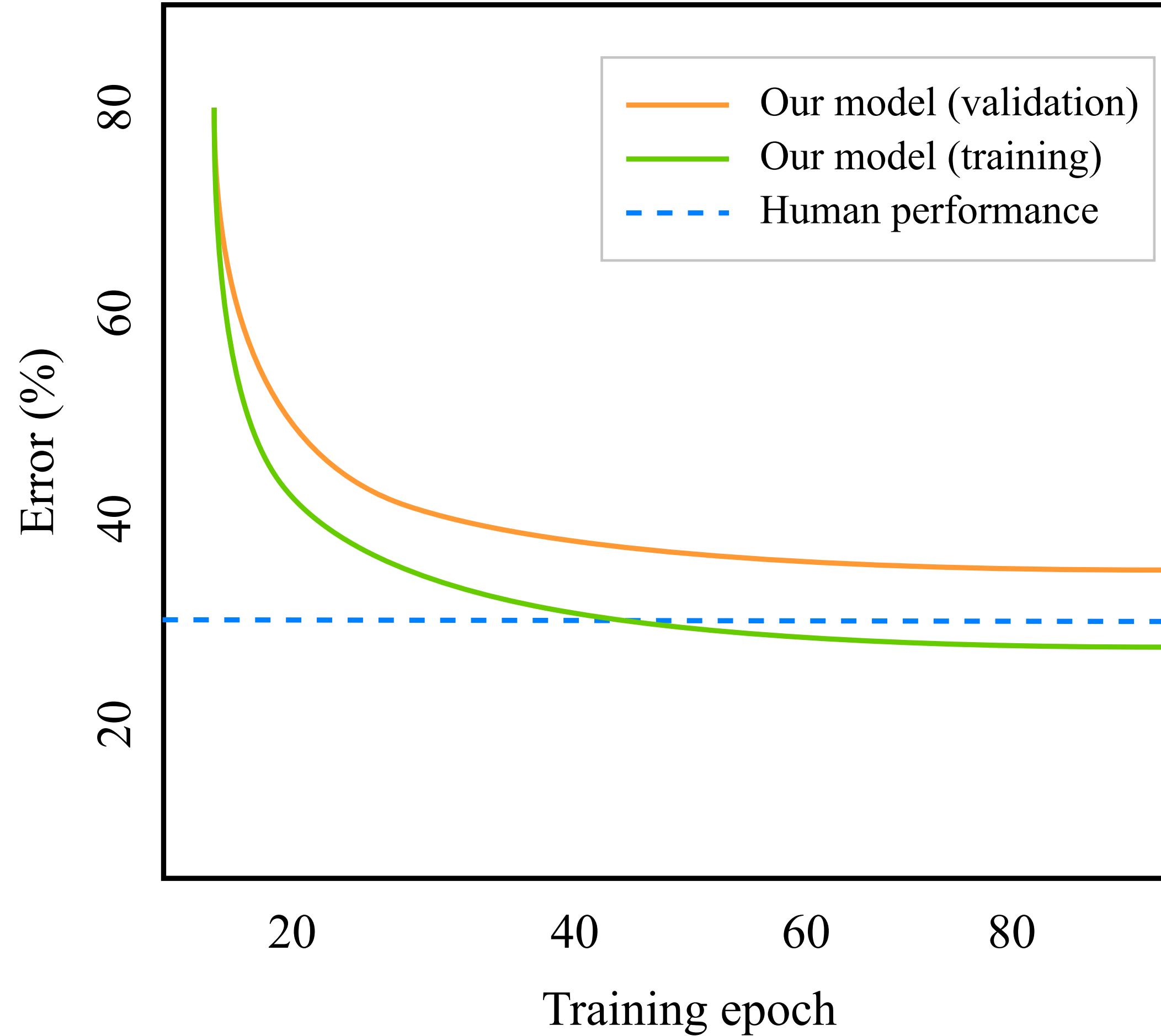
- Décomposition des données en 3 sous-ensembles

- Sous-ensemble d'entraînement pour apprendre le modèle
- Sous-ensemble de validation pour sélectionner le modèle
- Sous-ensemble de test pour évaluer le modèle

- En pratique: 2 sous-ensembles (entraînement et test) + validation croisée sur le sous-ensemble d'entraînement



# Définir le niveau de performance attendu



Avoir une idée du moment où on peut arrêter d'essayer d'améliorer le modèle.

# Identifier les métriques

## Classification

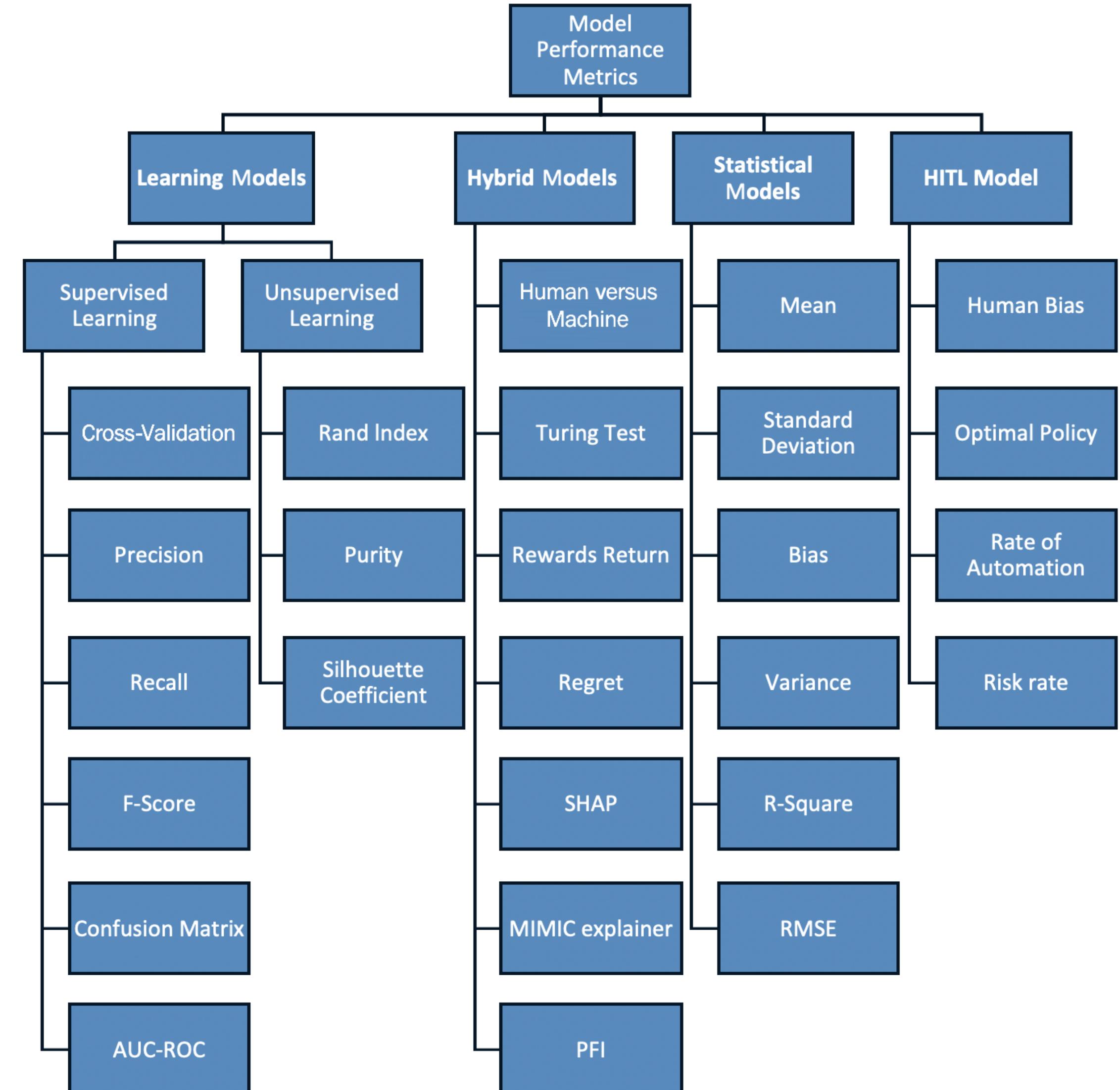
	spam (predicted)	not_spam (predicted)
spam (actual)	23 (TP)	1 (FN)
not_spam (actual)	12 (FP)	556 (TN)

$$\text{precision} \stackrel{\text{def}}{=} \frac{\text{TP}}{\text{TP} + \text{FP}}.$$

$$\text{recall} \stackrel{\text{def}}{=} \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$F_1 = \left( \frac{2}{\text{recall}^{-1} + \text{precision}^{-1}} \right) = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$$

$$\text{accuracy} \stackrel{\text{def}}{=} \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$



# Construction du programme d'entraînement

- Schéma de principe

- ▶ Boucle répétée sur les données d'entraînement
- ▶ Décomposition en plusieurs batchs
- ▶ Calcul des sorties
- ▶ Calcul de fonction perte (sorties réels vs sortie prédictes)
- ▶ Calcul des gradients pour mettre à jour les paramètres

```
for epoch in range(10): # loop over the dataset multiple times
    running_loss = 0.0

    for i, data in enumerate(trainloader, 0):
        # get the inputs; data is a list of [inputs, labels]
        inputs, labels = data

        # zero the parameter gradients
        optimizer.zero_grad()

        # forward + backward + optimize
        outputs = model(inputs)
        loss = criterion(outputs, labels)

        loss.backward()
        optimizer.step()

        # print statistics
        running_loss += loss.item()

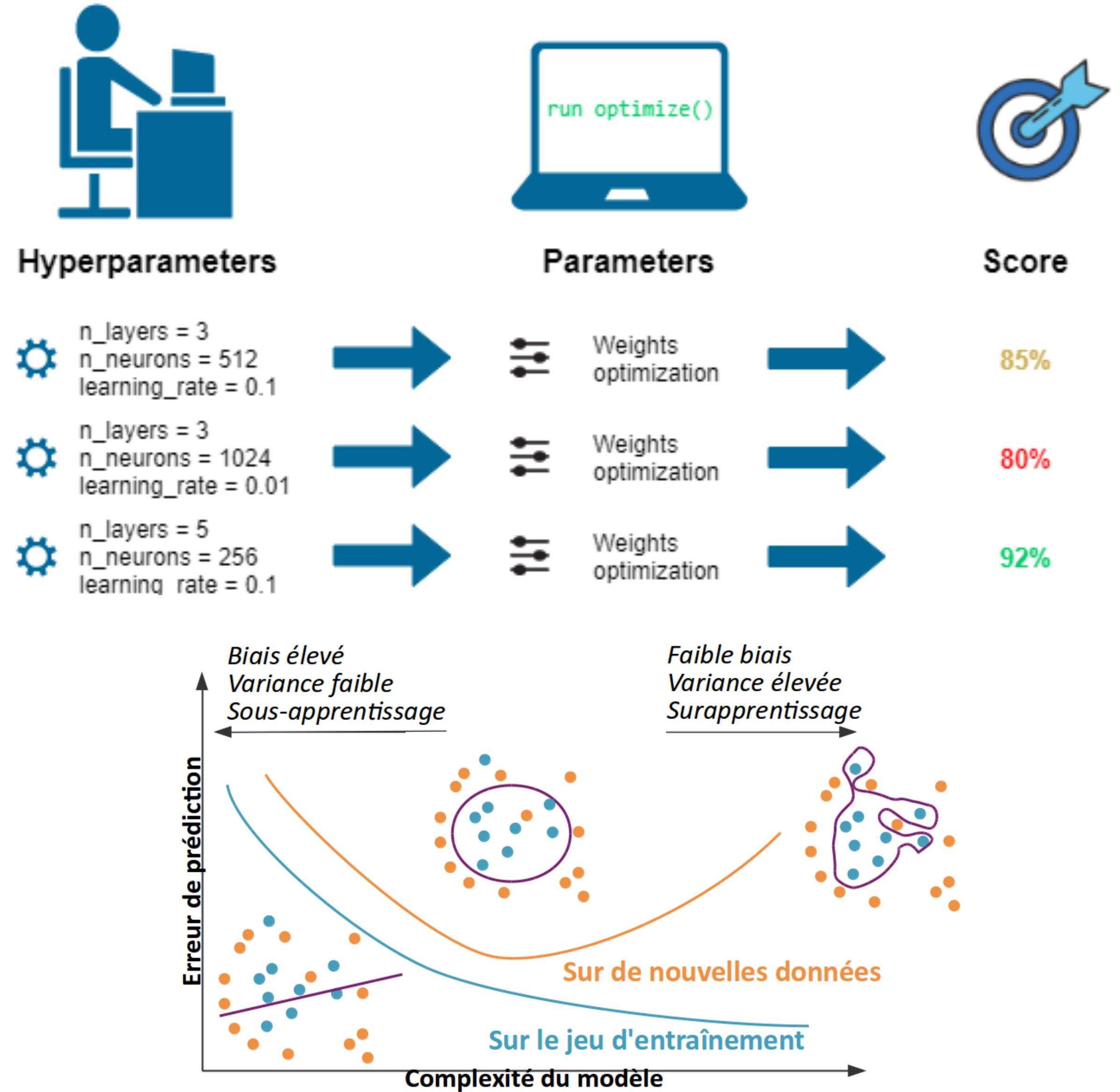
        if i % 2000 == 1999: # print every 2000 mini-batches
            print('[%d, %5d] loss: %.3f' %
                  (epoch + 1, i + 1, running_loss / 2000))

        running_loss = 0.0

    • print('Finished Training')
```

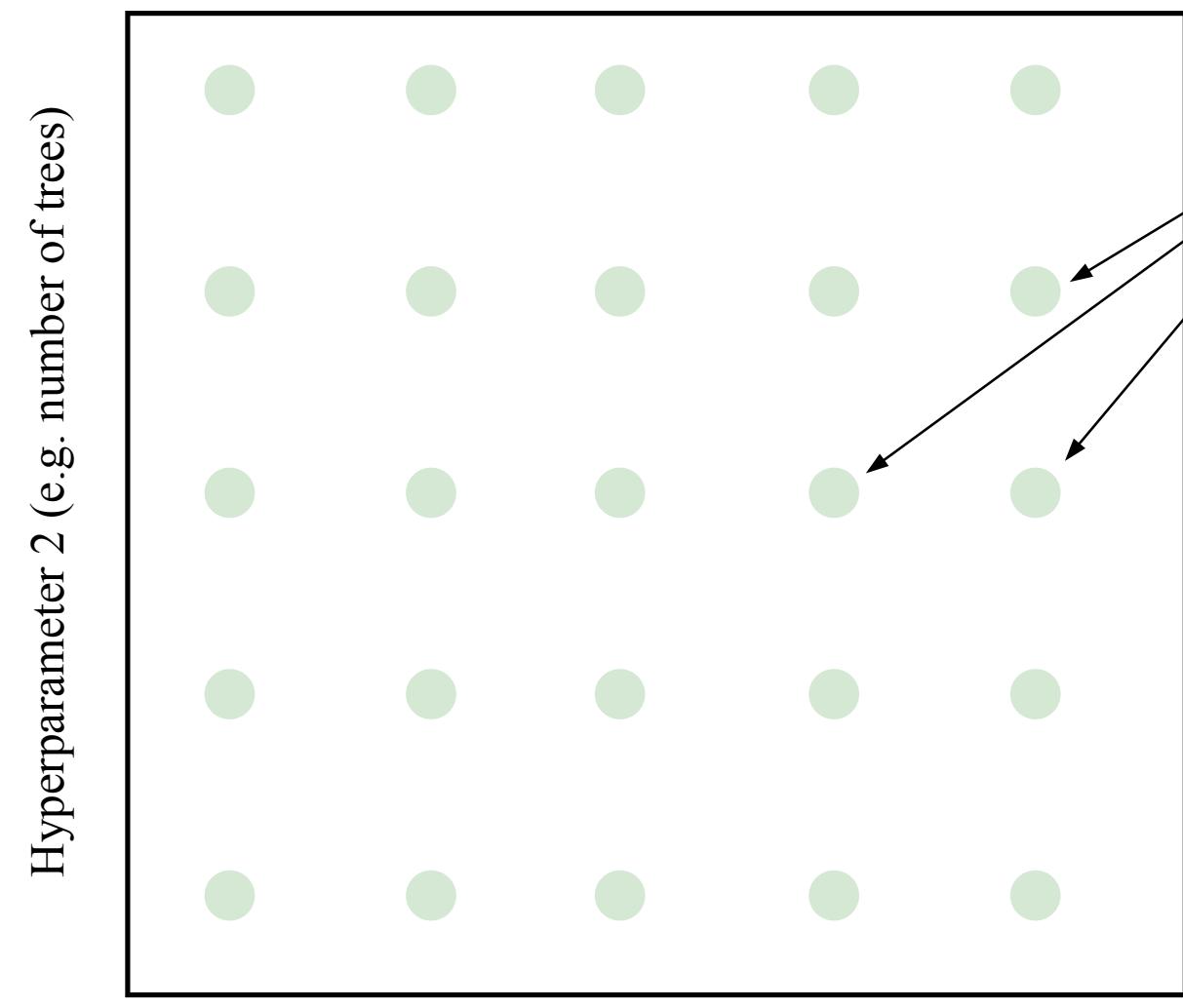
# Règlage des hyperparamètres

- Les hyperparamètres jouent un rôle important dans le processus d'apprentissage du modèle
- Certains hyperparamètres influencent la vitesse d'apprentissage, mais les hyperparamètres les plus importants contrôlent les deux compromis : biais-variance et précision-rappel
- Role de l'analyste de données de "régler" les hyperparamètres
- Chaque modèle d'apprentissage automatique et chaque algorithme d'apprentissage possède un ensemble unique d'hyperparamètres
- Techniques de régularisation, abandon, arrêt précoce et normalisation par lots

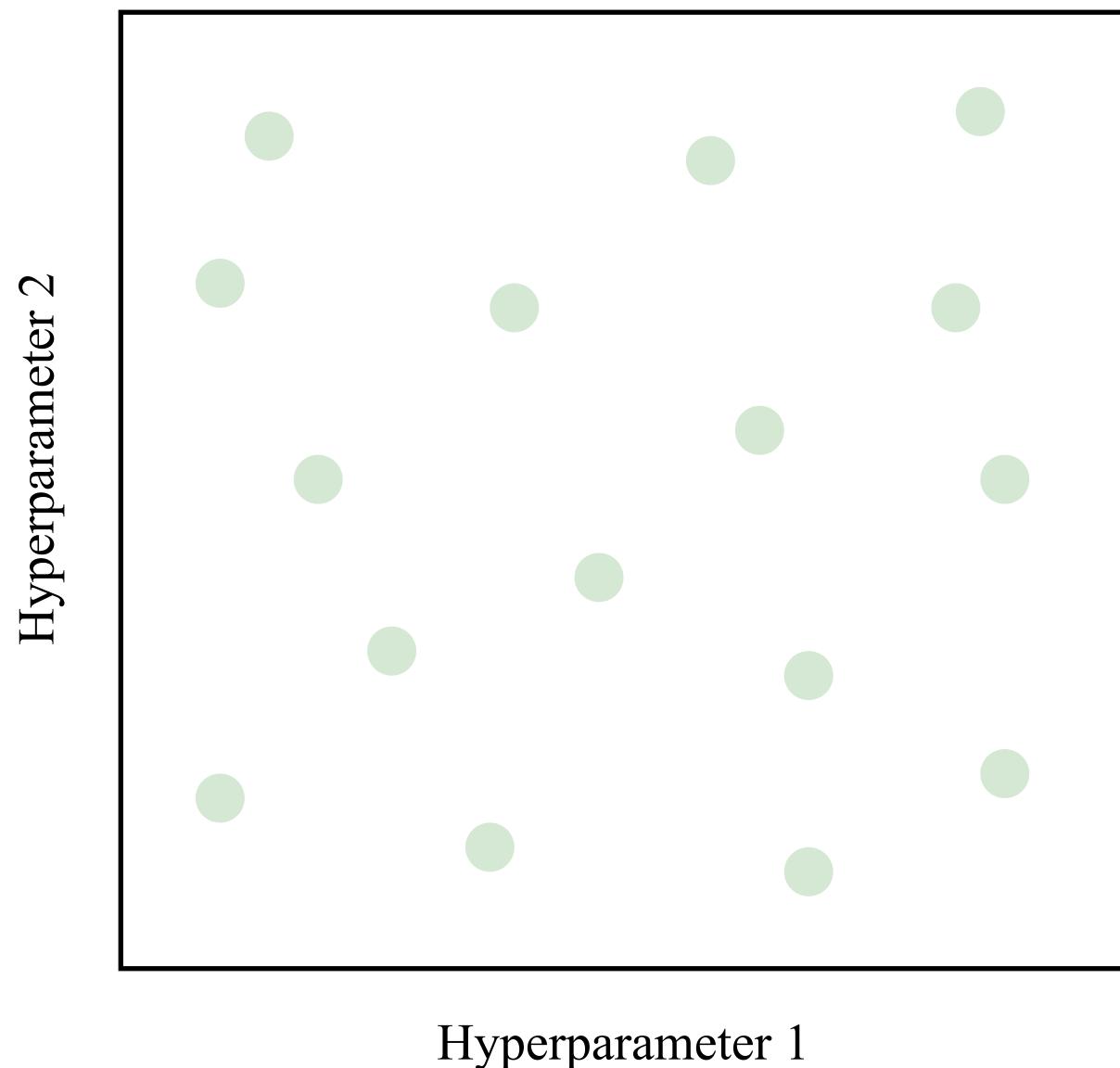


# Réglage des hyperparamètres

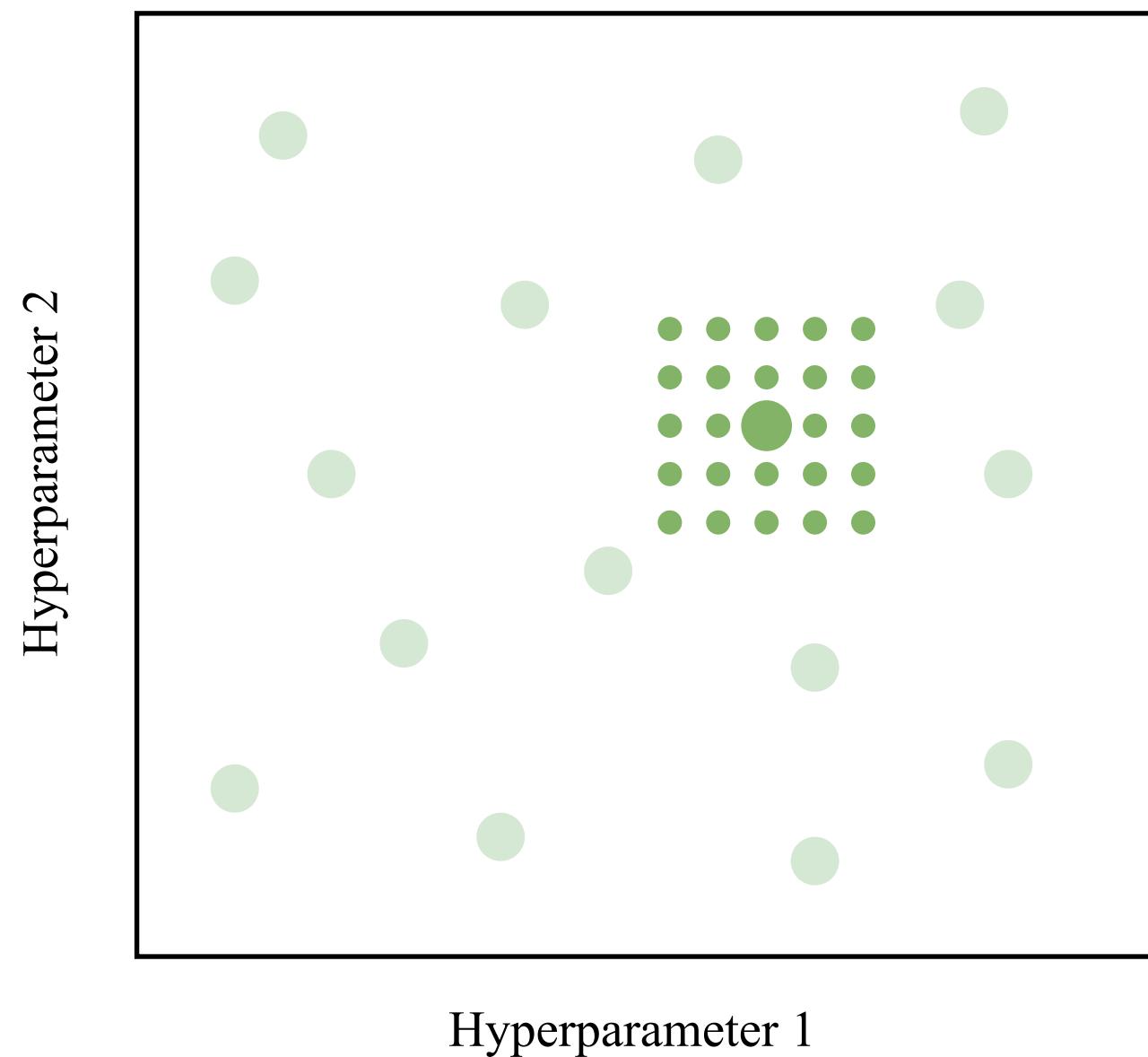
Recherche dans un grille



Recherche aléatoire



Recherche hybride



# Un peu de méthode

1. Définir une métrique de performance P
2. Présélectionner les algorithmes d'apprentissage
3. Choisir une stratégie de réglage des hyperparamètres T
4. Choisir un algorithme d'apprentissage A
5. Choisir une combinaison H de valeurs d'hyperparamètres pour l'algorithme A en utilisant la stratégie T
6. Utiliser l'ensemble d'apprentissage et former un modèle M en utilisant l'algorithme A paramétré avec des valeurs d'hyperparamètres H.
7. Utiliser l'ensemble de validation et calculer la valeur de la métrique P pour le modèle M
8. Décidez :
  - a. S'il reste des valeurs d'hyperparamètres non testées, choisissez une autre combinaison H de valeurs d'hyperparamètres en utilisant la stratégie T et retournez à l'étape 6.
  - b. Sinon, choisissez un algorithme d'apprentissage A différent et retournez à l'étape 5, ou passez à l'étape 9 s'il n'y a plus d'algorithmes d'apprentissage à essayer.
9. Retourner le modèle pour lequel la valeur de la métrique P est maximisée.

# Raffinement itératif du modèle

- Entraînez le modèle en utilisant les meilleures valeurs des hyperparamètres identifiés jusqu'à présent.
- Tester le modèle en l'appliquant à un petit sous-ensemble de l'ensemble de validation (100-300 exemples).
- Trouver les exemples d'erreurs les plus fréquents sur ce petit ensemble de validation
- Retirez les exemples de l'ensemble de validation, car votre modèle s'y adaptera trop
- Générer de nouvelles caractéristiques, ou ajouter plus de données d'entraînement pour corriger les modèles d'erreur observés
- Répétez jusqu'à ce qu'aucun modèle d'erreurs fréquent ne soit observé
  - erreurs uniformes apparaissant avec le même taux dans tous les cas d'utilisation
  - erreurs ciblées apparaissent plus fréquemment dans certains types de cas d'utilisation

# Raisons d'un mauvais comportement du modèle

- Mauvais comportement sur les données d'apprentissage
  - L'architecture du modèle ou l'algorithme d'apprentissage ne sont pas assez expressifs
  - Il y a trop de régularisation : réduire la régularisation
  - Les valeurs pour les hyperparamètres sont sous-optimales : utiliser une méthode de réglage automatique
  - Les caractéristiques conçues n'ont pas un pouvoir prédictif suffisant : besoin de nouvelles caractéristiques plus informative
  - Volume de données insuffisants pour que le modèle puisse généraliser : obtenir plus de données, utiliser l'augmentation des données ou l'apprentissage par transfert
  - Bug dans le code qui définit et entraîne le modèle
- Bon comportement sur les données d'apprentissage mais mauvais comportement sur les données de tests
  - Le modèle est sous-régularisé : ajouter une régularisation ou, pour les réseaux neuronaux, à la fois une régularisation et une normalisation par lots
  - la distribution de vos données d'apprentissage est différente de celle des données d'attente : réduisez le décalage de la distribution

# Propriétés du modèle

- Correction
- Robustesse
  - La robustesse d'un modèle d'apprentissage automatique fait référence à la stabilité de la performance du modèle après l'ajout d'un certain bruit aux données d'entrée.
  - Pour un modèle robuste, si l'exemple d'entrée est perturbé par l'ajout d'un bruit aléatoire, la performance du modèle se dégrade proportionnellement au niveau de bruit.
- Équité
  - L'équité signifie que chaque classe obtient une prédiction positive du modèle à un taux égal, en supposant que les exemples de ce groupe remplissent les conditions requises
  - Certains attributs sont sensibles et doivent être protégés de l'injustice
  - L'exclusion des attributs protégés du vecteur de caractéristiques dans les données d'apprentissage ne garantit pas que le modèle aura une parité démographique, car certaines des caractéristiques restantes peuvent être corrélées avec celles qui sont exclues.

# Stockage des modèles

- Besoins
  - ▶ Mutualiser les modèles performants
  - ▶ Éviter des entraînements répétitifs coûteux en temps et en mémoire

- Magasin de modèles
  - ▶ Stockage des modèles dans différents formats et prêt à l'emploi pour la mise en production (package)
  - ▶ Gestion des métadonnées sur les modèles
    - données ayant servi à l'entraînement et aux tests
    - date de création, versions ...
    - résultats obtenus, frameworks utilisés

The screenshot displays two main interfaces related to model management:

**Microsoft Azure Machine Learning Studio:** This interface shows a model named "support-vector-classifier:1". The "Details" tab is selected, displaying attributes like Version 1, ID support-vector-classifier:1, and Date registered 2019-10-16. It also shows sections for Artifacts, Endpoints, Explanations (preview), Fairness (preview), and Datasets. A sidebar on the left lists options such as New, Home, Author, Notebooks, Automated ML (preview), Designer (preview), Assets, Datasets, Experiments, Pipelines, Models (which is selected), Endpoints, Manage, Compute, Datastores, and Data Labeling.

**mlflow:** This interface shows a registered model named "support-vector-classifier:1". It includes a "Default" run section with details like Date: 2019-10-16 22:49:25, Source: train\_predict.py, User: sid.murching, Duration: 6.2s, and Notes (None). It also shows Parameters, Metrics, Tags (empty), and Artifacts. The artifacts section shows a directory structure: model/MLmodel/conda.yaml/tfmodel. A "Register Model" button is visible.

**Registered Models:** This table lists registered models. The columns are Name, Latest Version, Staging (highlighted in orange), Production (highlighted in green), and Last Modified. Two models are listed: Model A (Version 1, Last Modified 2019-10-16 22:51:19) and Model B (Version 1, Last Modified 2019-10-16 22:51:52).

# Déploiement, surveillance et maintenance

# Déploiement du modèle

- Déployer un modèle (testé de manière approfondie) signifie le rendre disponible pour accepter des requêtes générées par les utilisateurs du système de production
- Code accompagnant le modèle : extracteur de caractéristiques et convertisseur de prédictions en format interprétable
- Une fois que le système de production accepte la requête, celle-ci est transformée en un vecteur de caractéristiques.
- Le vecteur de caractéristiques est ensuite envoyé au modèle comme entrée pour le scoring. Le résultat est alors renvoyé dans un format compréhensible à l'utilisateur
- Déploiement sur un serveur ou sur un appareil de l'utilisateur (pour tous les utilisateurs ou une fraction)
- Type de déploiements:
  - De manière statique, dans le cadre d'un logiciel installable
  - De manière dynamique sur le dispositif de l'utilisateur
  - Dynamiquement sur un serveur
  - Via le streaming de modèles

# Déploiement dynamique<sup>1</sup>

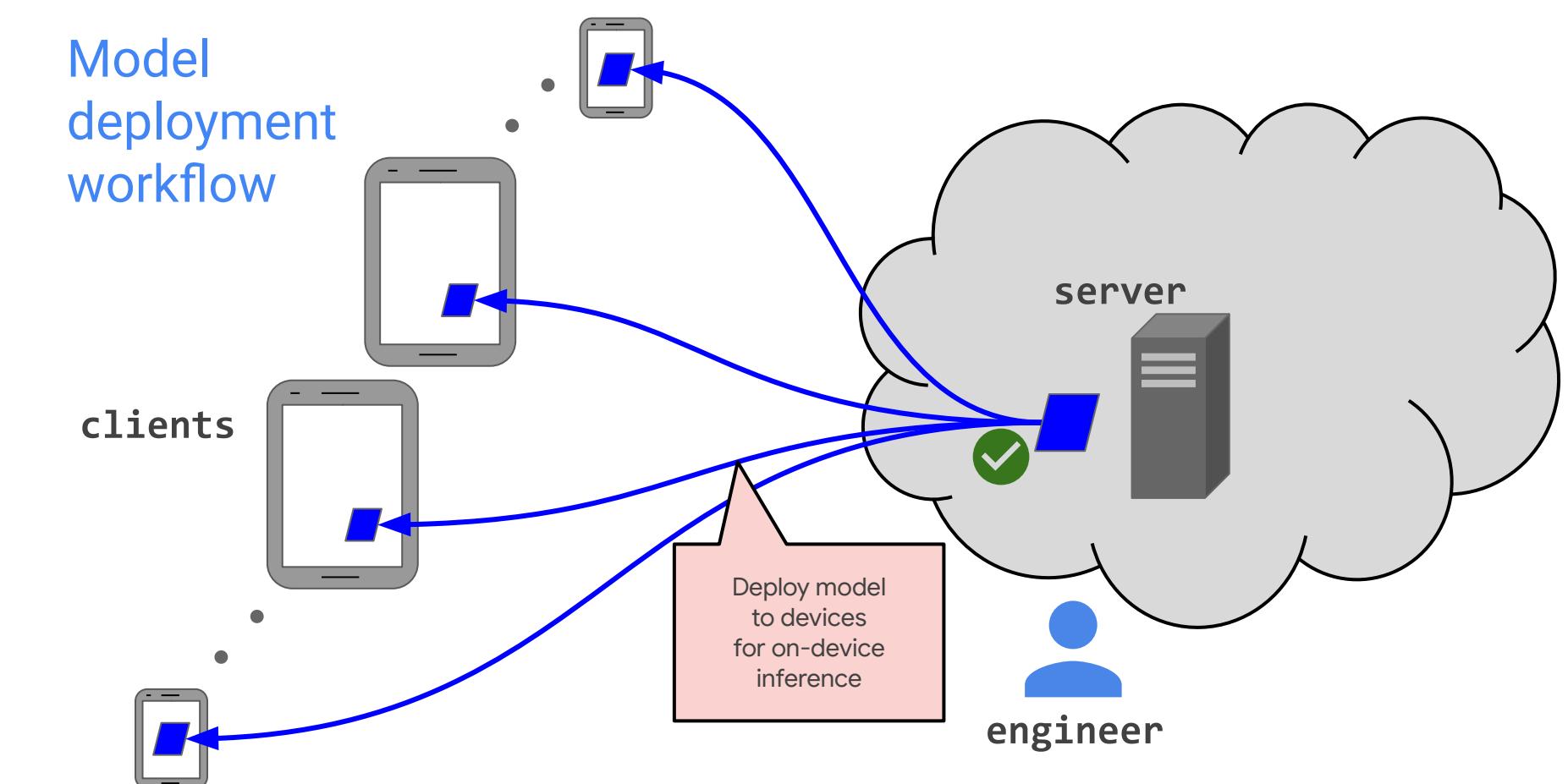


TensorFlow Lite

- **Cas 1: Dispositif de l'utilisateur**
- Le modèle ne fait pas partie du code binaire de l'application
- Les mises à jour du modèle sont effectuées sans mettre à jour l'application
- Possibilité de sélectionner le modèle en fonction du profil utilisateurs ou des ressources du dispositif
- Le fichier du modèle ne contient que les paramètres appris tandis que le dispositif de l'utilisateur possède un environnement d'exécution pour le modèle
- Avantages: accès rapide au modèle, prédiction en local
- Inconvénients: bande passante, coût de démarrage, infrastructure, difficulté du suivi des performances du modèles
- Boucle de feedback pour contrôler les performances



TensorFlow.js

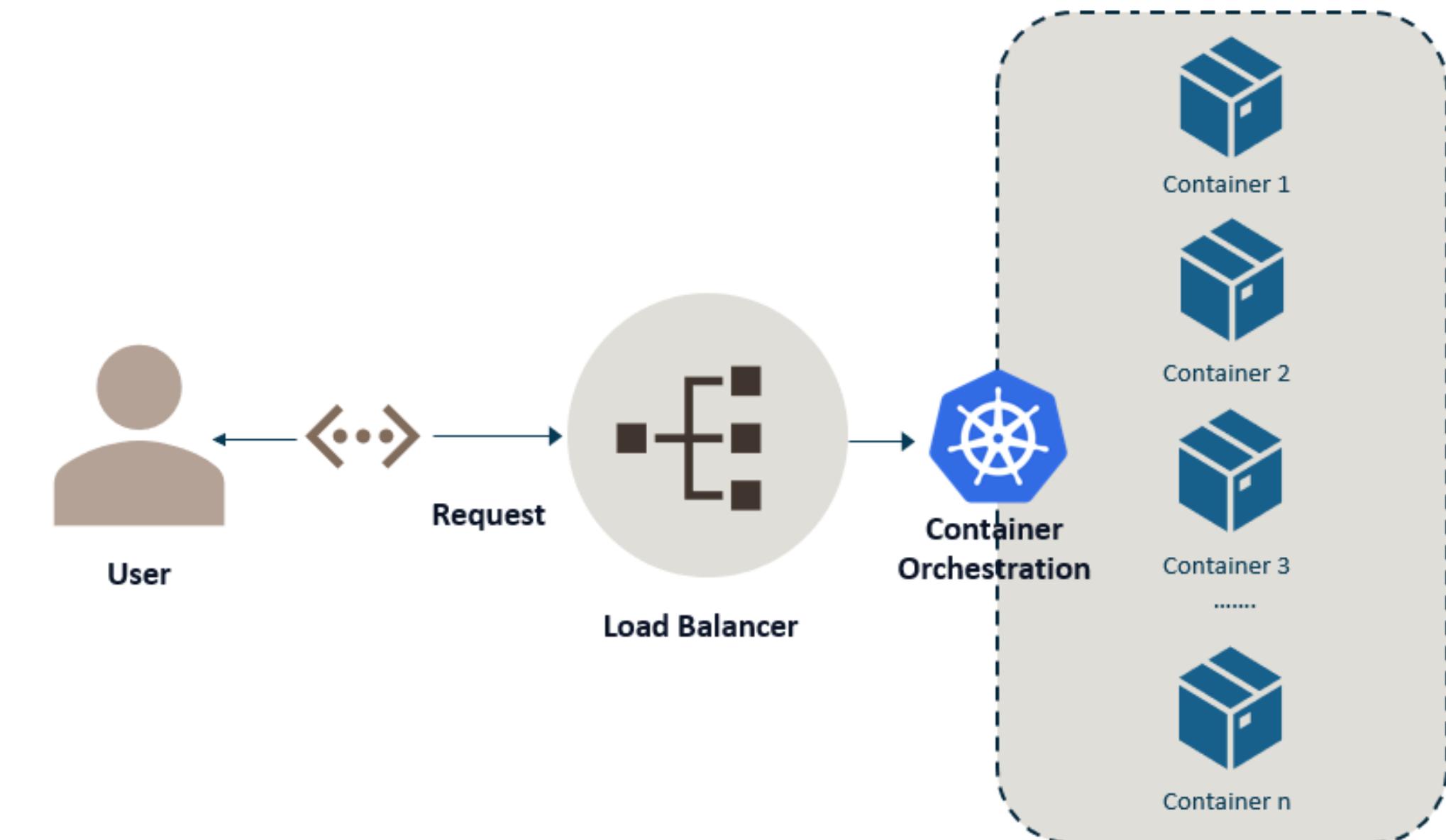


# Déploiement dynamique<sup>2</sup>

- Cas 2: Serveur
- Modèle de déploiement dynamique le plus fréquent
- Mise à disposition sous la forme d'une API REST (service Web) en utilisant un framework applicatif
- Déploiement dans une infrastructure
  - Régulateur de charge
  - Machines virtuelles, conteneurs (docker, kubernetes), serverless (AWS lambda, Microsoft Function,...)

## Exemple de Réponse : Microsoft Detection API

```
{  
  "rectangle":{  
    "x":523,  
    "y":377,  
    "w":185,  
    "h":46  
  },  
  "object":"computer keyboard",  
  "confidence":0.51  
},  
]  
,"requestId":"a7fde8fd-cc18-4f5f-99d3-897dcd07b308",  
"metadata":{  
  "width":1260,  
  "height":473,  
  "format":"Jpeg"  
}
```

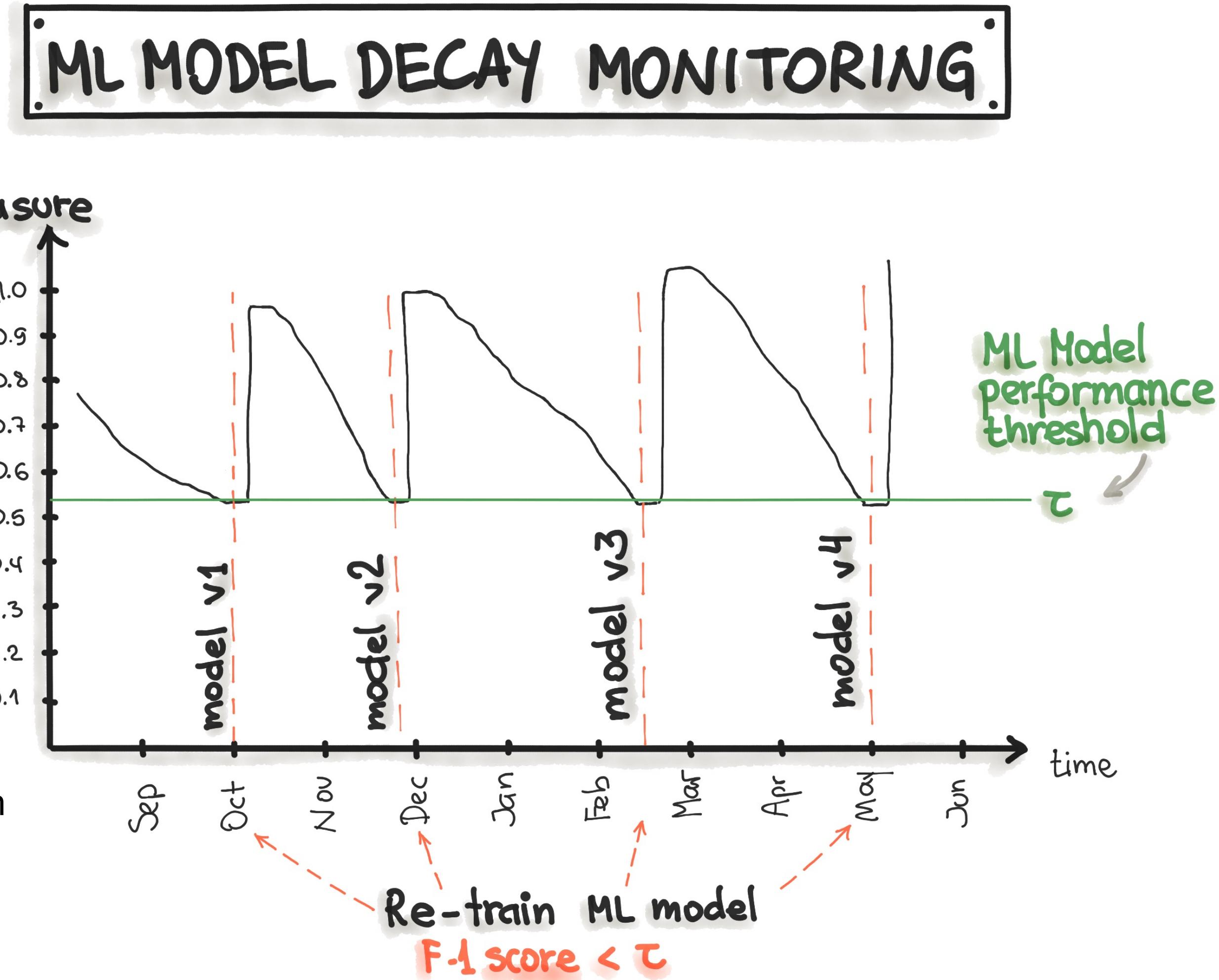


# Stratégies de déploiement

- Déploiement unique
  - Sérialisation du nouveau modèle dans un fichier et remplacement dans le service (Arrêt/relance du service avec éventuellement construction d'un nouveau conteneur)
  - Simple à mettre en oeuvre mais risquée en cas de bogue dans le modèle ou le code associé
- Déploiement silencieux
  - Conservation de l'ancienne version et exécution en parallèle de la nouvelle version
  - Pas d'impact sur les utilisateurs et comparaison des performances mais coûteux en terme de ressources et pas toujours possible si besoin de feedback utilisateurs
- Déploiement Canari
  - Mise à disposition de la nouvelle version pour un petit nombre d'utilisateurs afin de valider les performances du nouveau modèle

# Surveillance

- Assurer que le modèle soit correctement servi et les performances restent dans les limites acceptables
- Risques
  - Les nouvelles données d'apprentissage utilisées pour mettre à jour le modèle ont dégradé ses performances
  - Les données réelles en production ont changé, mais pas le modèle
  - Le code d'extraction des caractéristiques a été mis à jour de manière significative, mais le modèle ne s'est pas adapté
  - une ressource nécessaire pour générer une caractéristique a changé ou est devenue indisponible
  - le modèle fait l'objet d'un abus ou d'une attaque adverse
- Utilisation d'un ensemble de test de confiance mis à jour avec de nouvelles données afin d'éviter tout décalage éventuel de la distribution
- Tester sur les exemples de l'ensemble de bout en bout
- Métrique pour mesurer l'évolution dans le temps : le biais de prédiction
  - distribution des classes prédites ~ distribution des classes observées



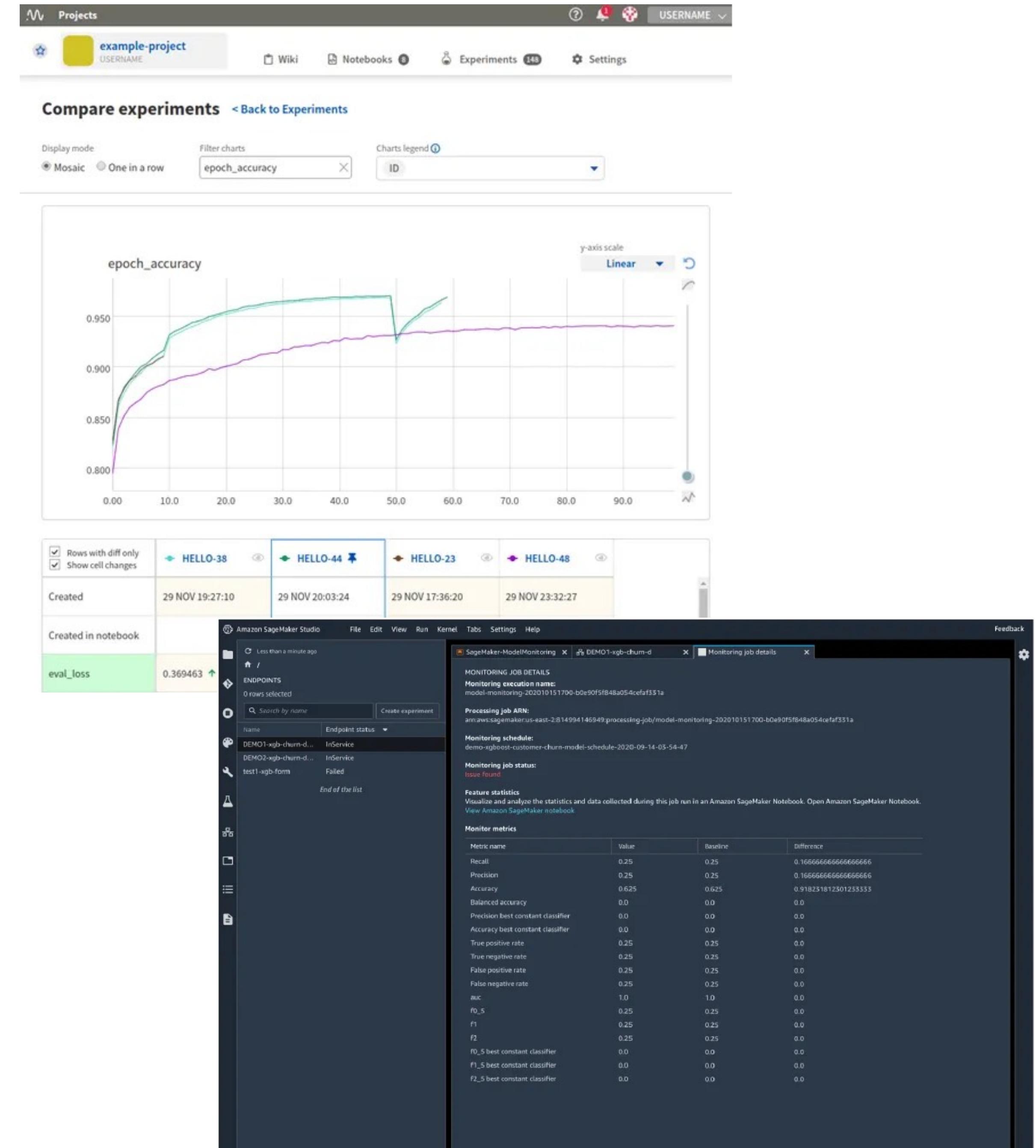
# Surveillance

- **Au niveau des données**

- Alimentation des colonnes liées à des sources extérieures
- Définition et format des données
- Distribution des valeurs des caractéristiques : détection par tests statistiques

- **Au niveau des prédictions**

- Valeurs de prédiction minimales et maximales,
- Valeurs de prédiction médiane, moyenne et écart-type sur une période donnée
- Latence lors de l'appel de l'API du modèle
- Consommation de mémoire et l'utilisation du CPU lors de l'exécution des prédictions
- Système d'alertes et journalisation des évènements sur les données et prédictions

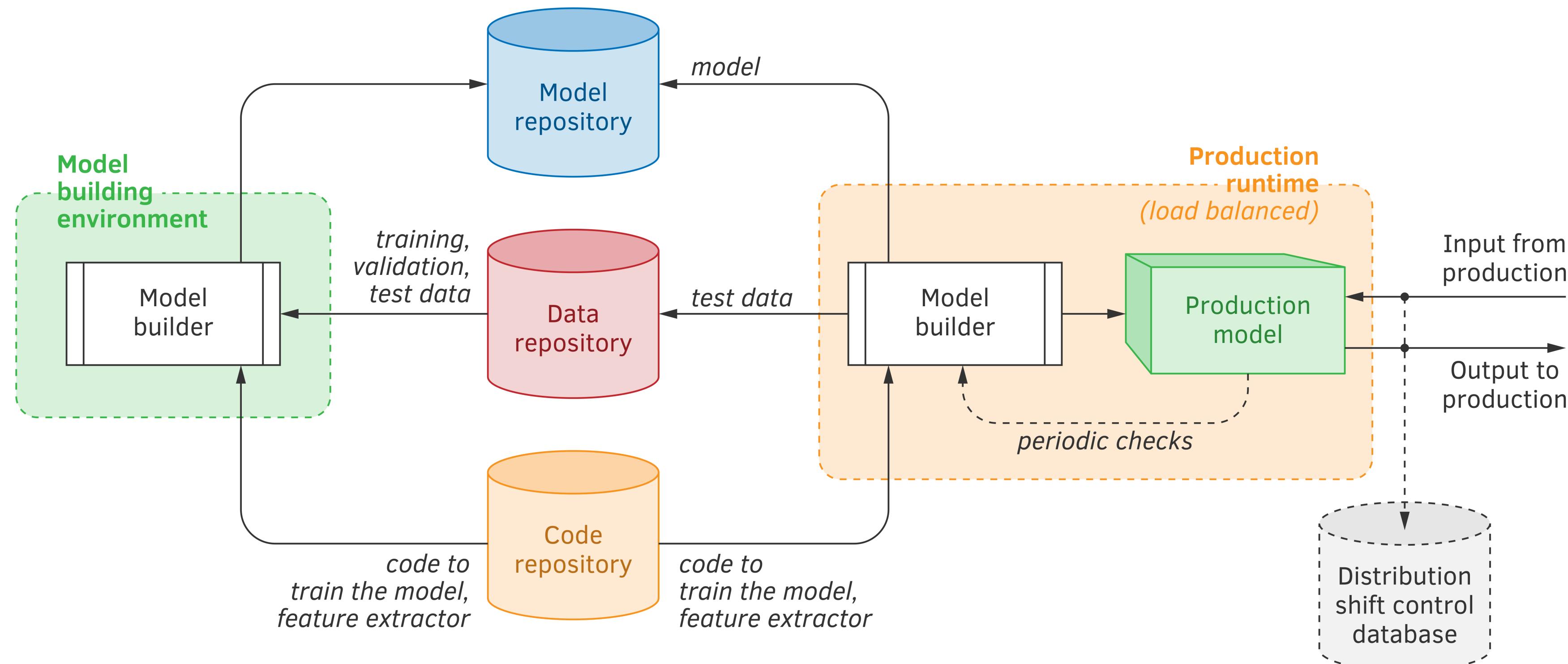


# Maintenance

- Rythme des mise à jour selon
  - la fréquence de ses erreurs et leur criticité
  - le degré de " fraîcheur " du modèle, pour qu'il soit utile
  - la vitesse à laquelle de nouvelles données d'entraînement deviennent disponibles
  - le temps qu'il faut pour ré-entraîner un modèle,
  - le coût du déploiement du modèle
  - la contribution d'une mise à jour du modèle au produit et à la réalisation des objectifs de l'utilisateur
- Rechercher la solidité au fur et à mesure des mises à jour (fréquent au début)

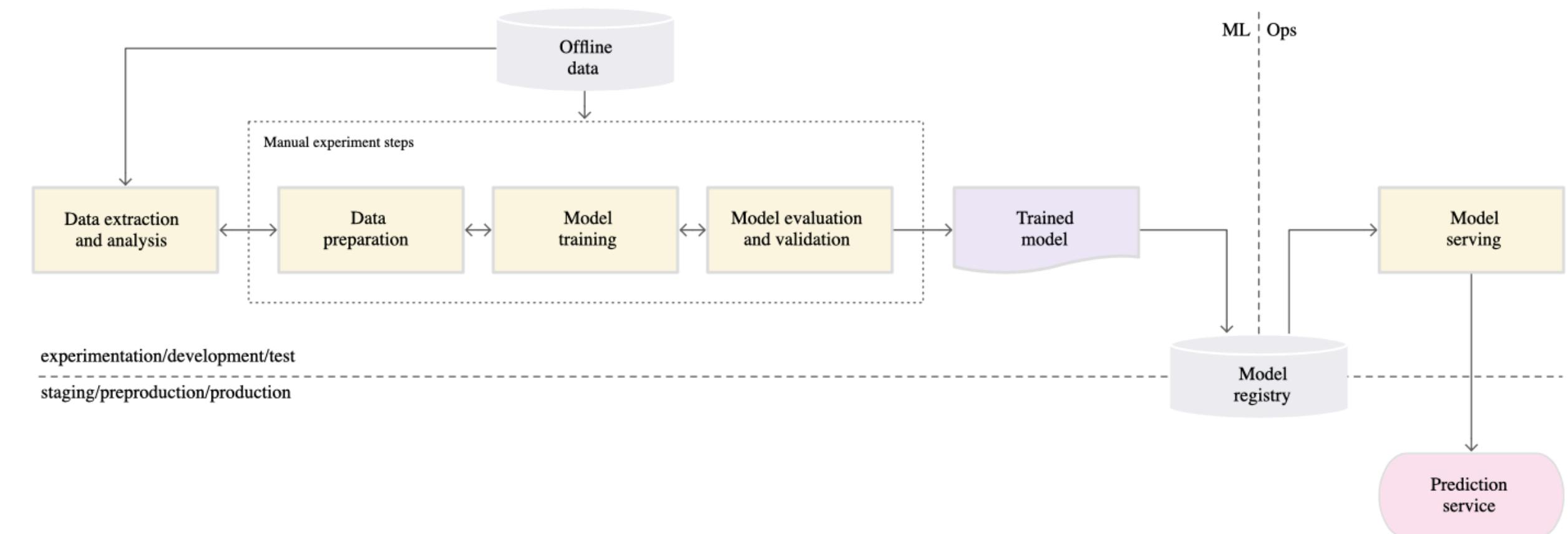
# Comment mettre à jour

- Utiliser une plateforme de déploiement et de maintenance automatique
  - ▶ 2 environnements: un pour l'entraînement, un pour l'exécution



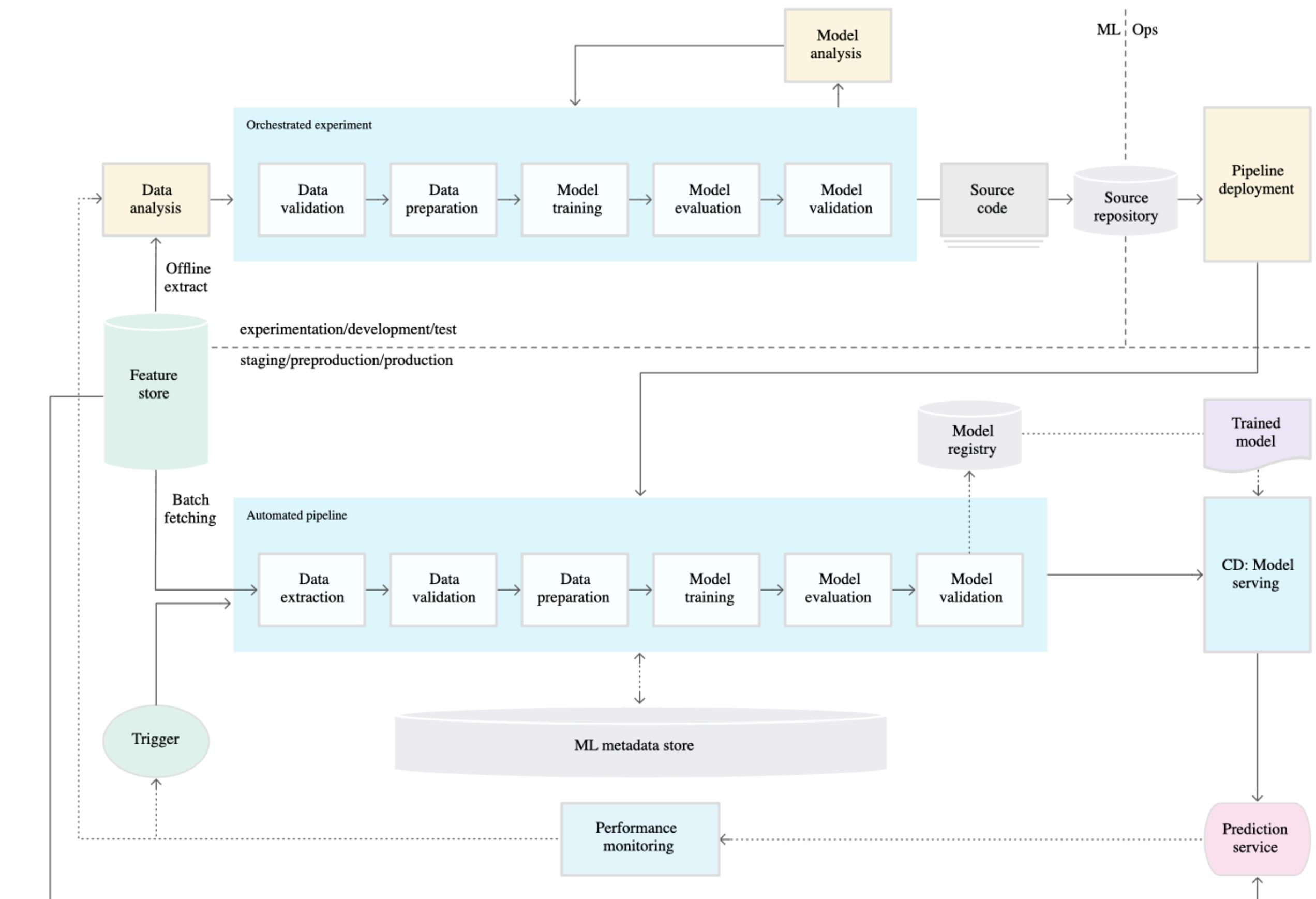
# Vers une pipeline de plus en plus automatisée

- **Niveau MLOps 0:** processus de création et de déploiement de modèles de ML entièrement manuel
  - Ce processus s'appuie sur un code expérimental écrit et exécuté dans des notebooks de manière interactive par un data scientist, jusqu'à l'obtention d'un modèle fonctionnel
  - Dissociation entre le ML et les opérations (transmission d'un modèle entraîné à l'équipe d'ingénieurs chargés de le déployer sur leur infrastructure d'API)
  - Itérations de version peu fréquentes
  - Pas d'intégration continue (test de code intégré aux scripts)
  - Pas de livraison continue
  - Absence de surveillance active des performances



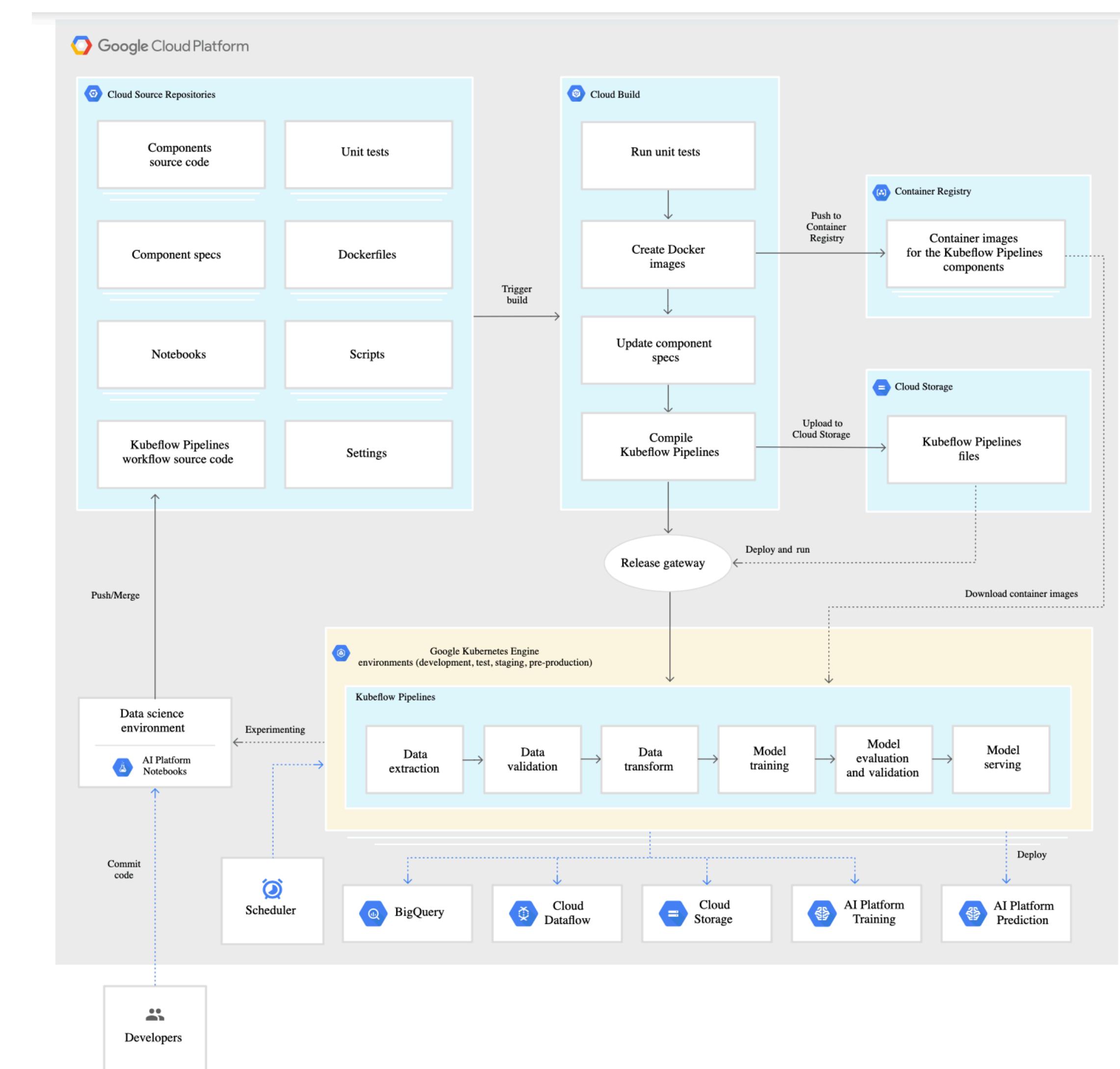
# Vers une pipeline de plus en plus automatisée

- Niveau MLOps 1: automatisation du pipeline de ML
- Orchestration des étapes de la pipeline de développement/tests
- Déploiement d'un pipeline d'entraînement complet qui s'exécute automatiquement et de manière récurrente
- Entrainement continu du modèle avec déclencheurs de pipeline
  - dégradation de performance
  - nouvelles données d'entraînement
  - changement dans la distribution des données

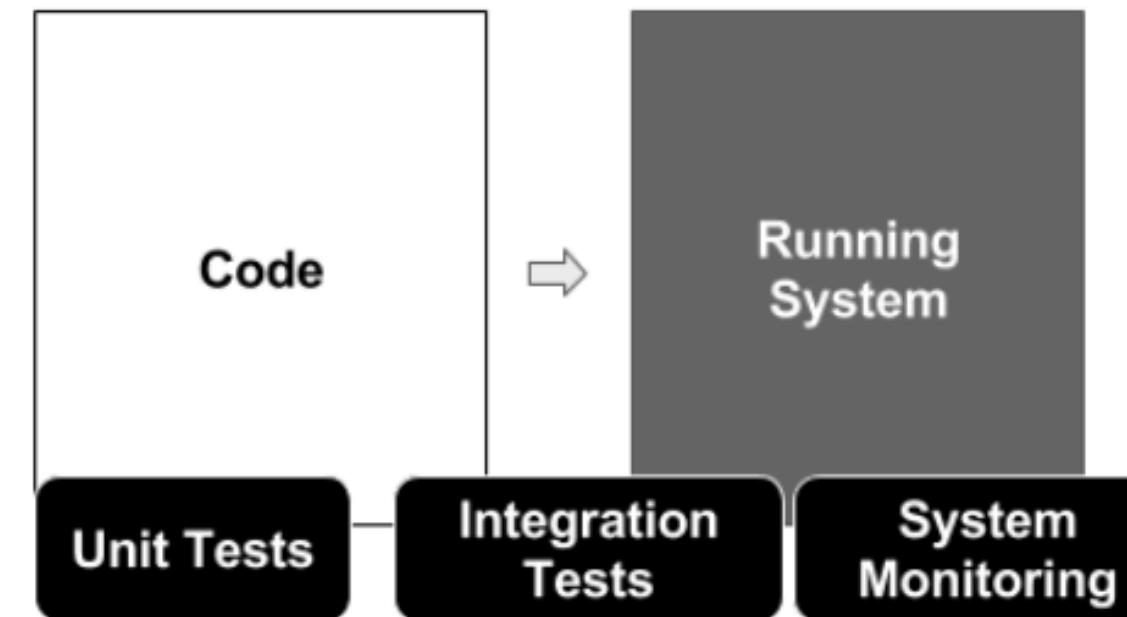
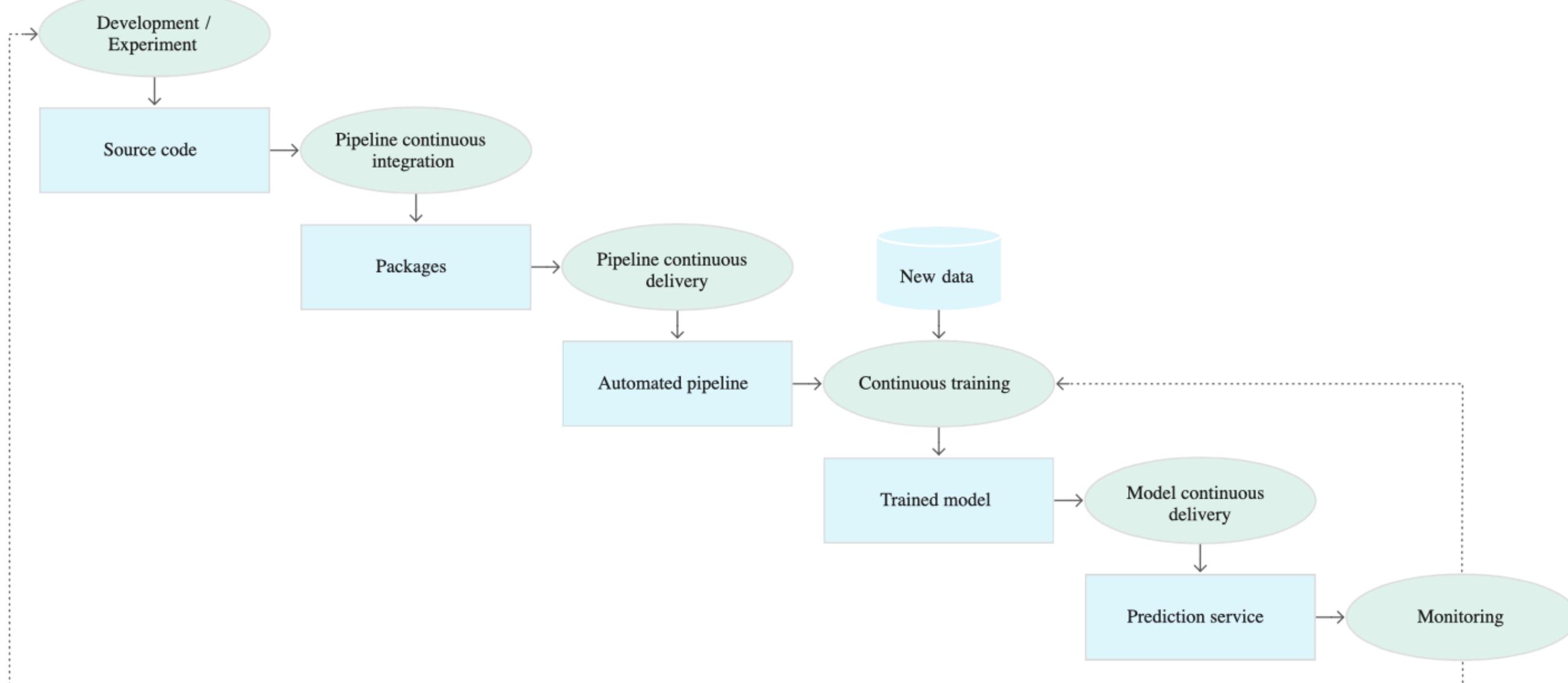


# Intégration, entraînement et livraison continue

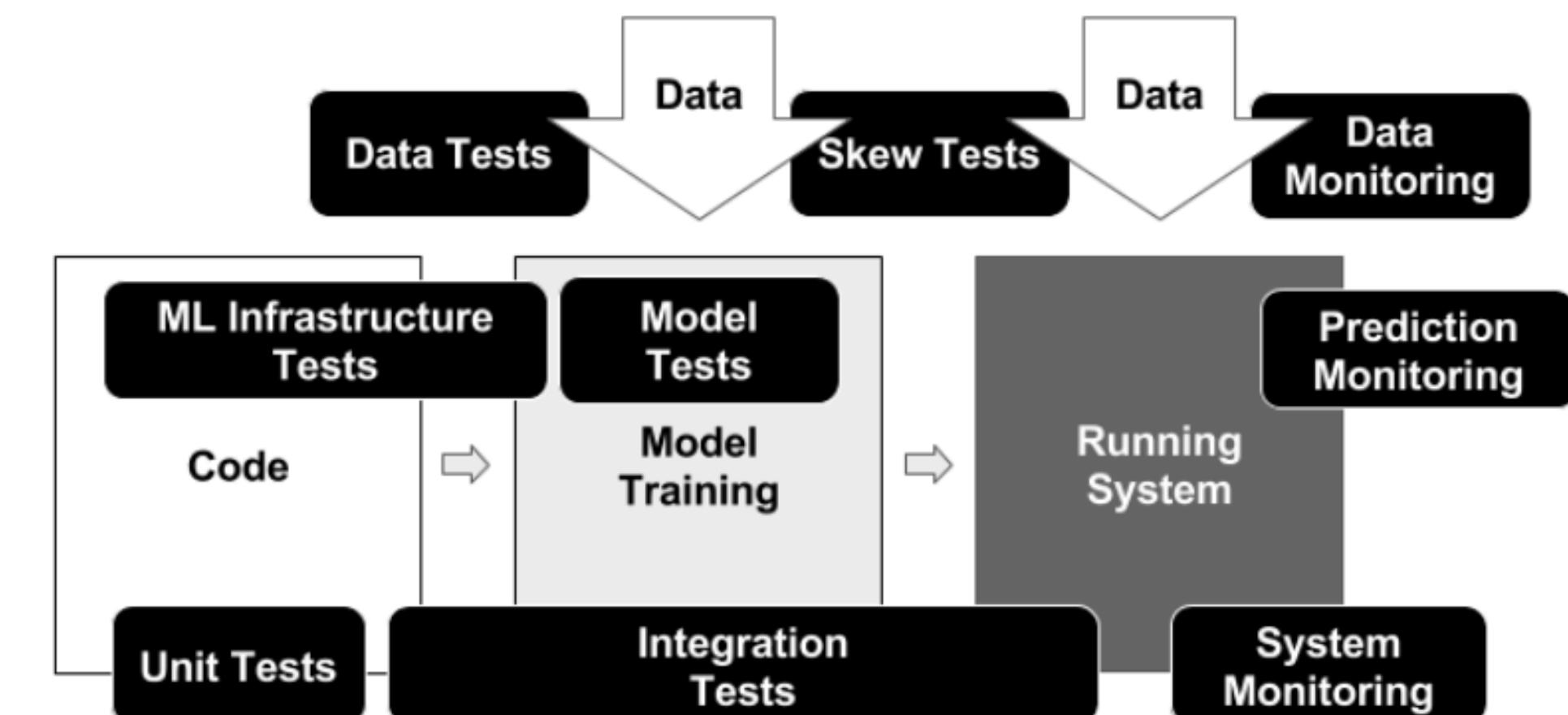
- **Niveau MLOps 2:** Automatisation de bout en bout, de l'ingestion des données à la livraison du service
- A l'arrivée de nouvelles données, les processus de validation des données, de réentraînement et de validation du modèle sont déclenchés pour un redéploiement
- Dans l'esprit de l'intégration continue pour le code
  - Tests visant le code: extraction de caractéristique, validation des données, communication entre composants de la pipeline
  - Tests visant le modèle: convergence de l'entraînement de votre modèle, conformité des prédictions
- Déploiement automatiquement d'un nouveau service de prédiction si les tests sont passées
- Surveillance continue des données de production et des mesures de performance du modèle, qui sont liées aux mesures de l'entreprise



# Integration, entraînement et livraison continue



Traditional System Testing and Monitoring



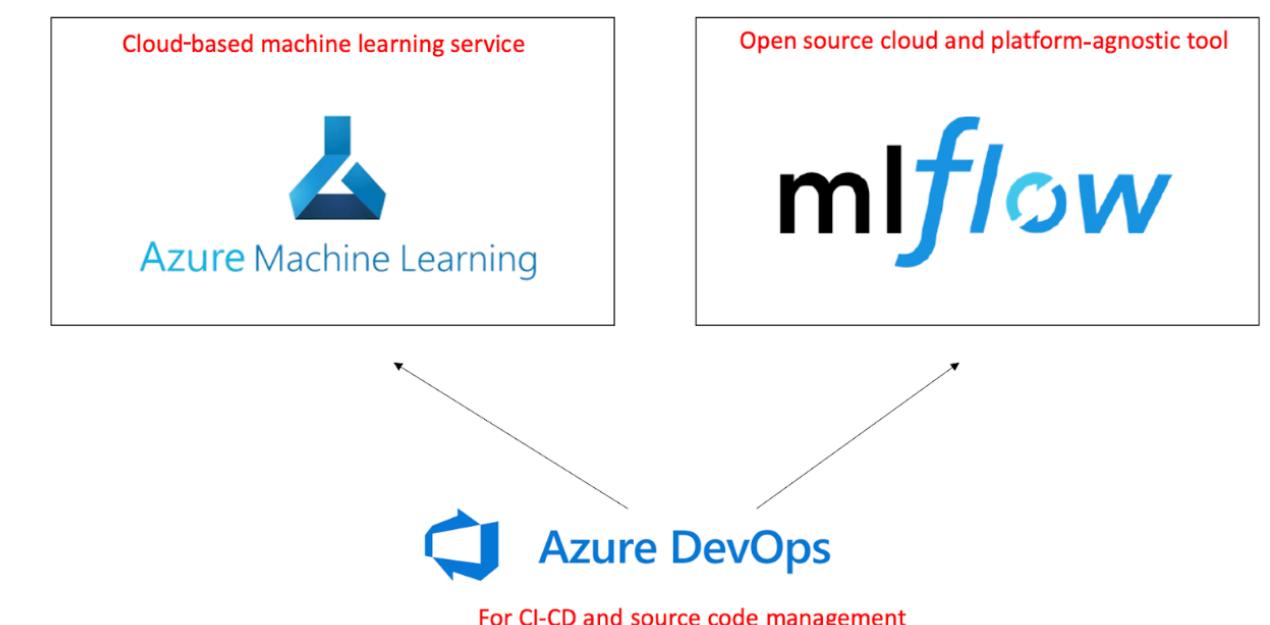
ML-Based System Testing and Monitoring

# **Exemple de plateformes**

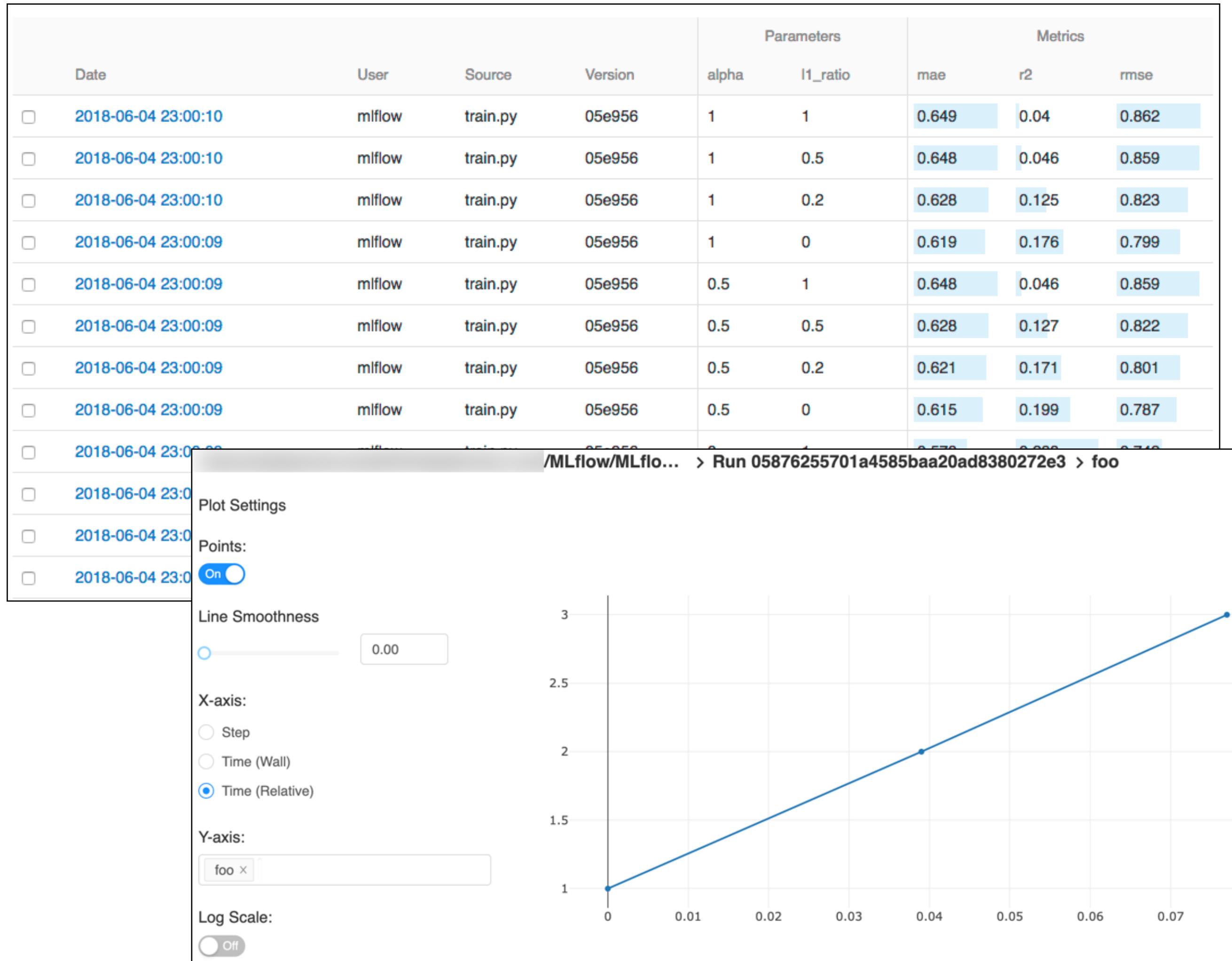
# MLflow<sup>1</sup>

- Plateforme open source permettant de gérer le cycle de vie de l'apprentissage automatique de bout en bout
- Suivi des expériences pour enregistrer et comparer les paramètres et les résultats (MLflow Tracking)
- Conditionnement du code ML sous une forme réutilisable et reproductible afin de le partager avec d'autres data scientists ou de le transférer en production (MLflow Projects)
- Gérer et déployer des modèles à partir d'une variété de bibliothèques ML vers une variété de plateformes de service et d'inférence de modèles (MLflow Models)
- Fournir un magasin central de modèles pour gérer de manière collaborative le cycle de vie complet d'un modèle MLflow, y compris les versions du modèle, les transitions d'étape et les annotations (MLflow Model Registry)

**mlflow**



# MLflow<sup>2</sup>



mlflow

Run 7c1a0d5c42844dcdb8f5191146925174

Experiment Name: Default Start Time: 2018-06-04 23:47:22

Source: train.py Git Commit: 3aa48cff58b8d9d69f5

User: mlflow Duration: 145ms

▼ Parameters

Name	Value
alpha	0
l1_ratio	0

▼ Metrics

Name	Value
mae	0.578
r2	0.288
rmse	0.742

► Tags

▼ Artifacts

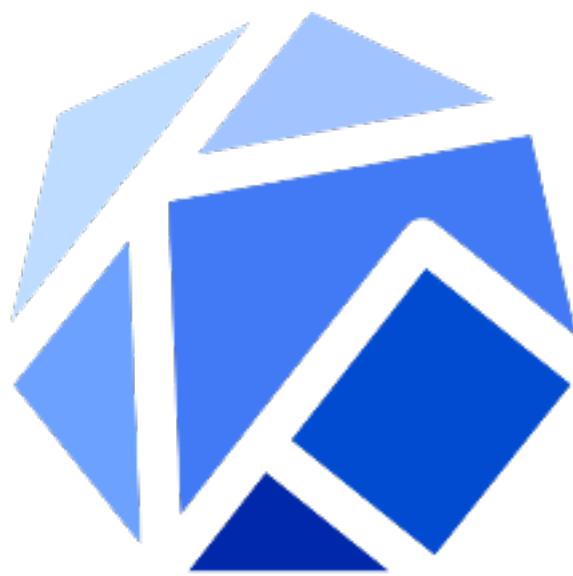
Full Path:/Users/mlflow/mlflow-prototype/mlruns/0/7c1a0d5c42844dcdb8f5191146925174/artifacts/model/MLmodel  
Size: 259B

```

artifact_path: model
flavors:
  python_function:
    data: model.pkl
    loader_module: mlflow.sklearn
sklearn:
  pickled_model: model.pkl
  sklearn_version: 0.19.1
run_id: 7c1a0d5c42844dcdb8f5191146925174
utc_time_created: '2018-06-05 06:47:22.757025'

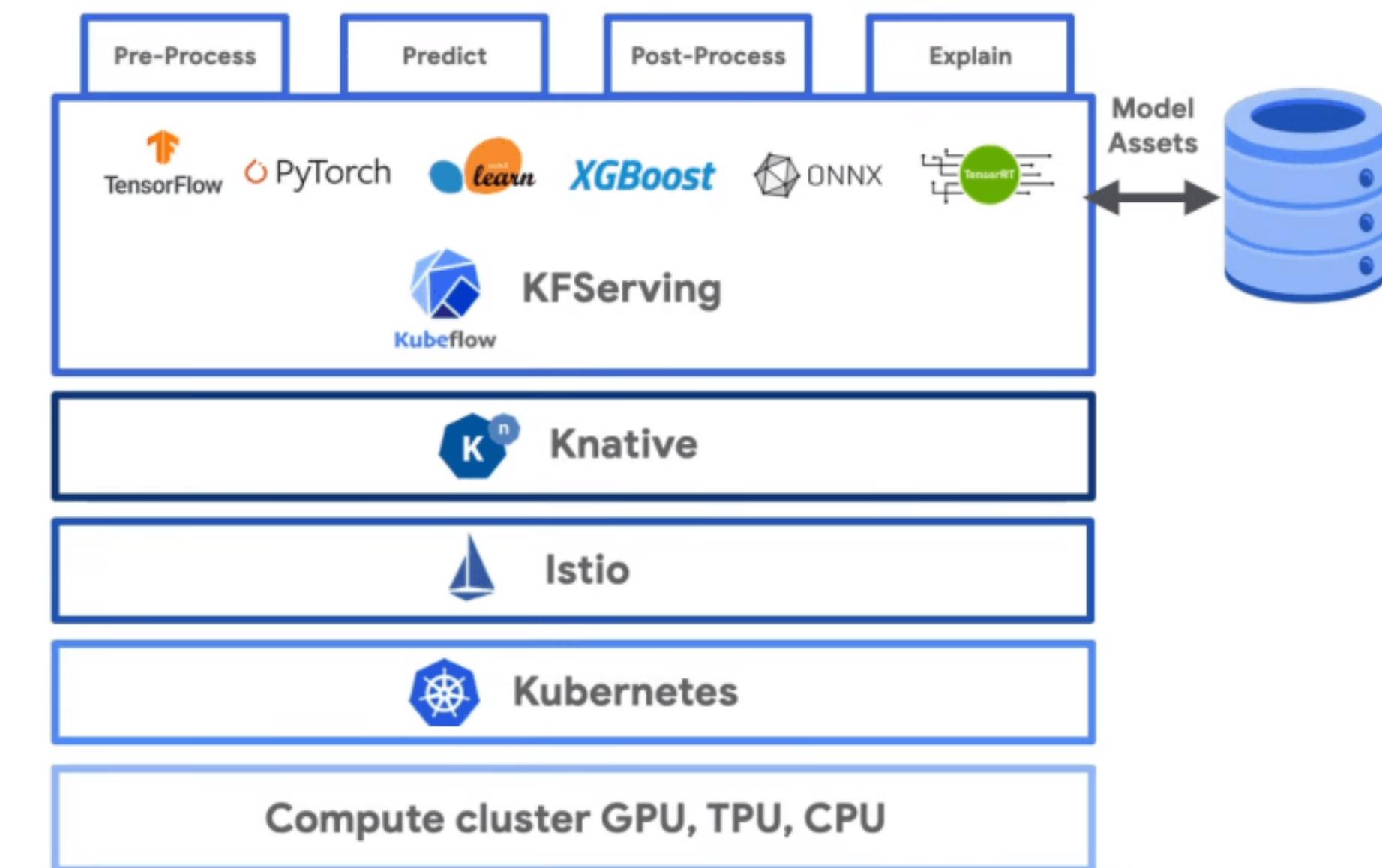
```

# Kubeflow<sup>1</sup>

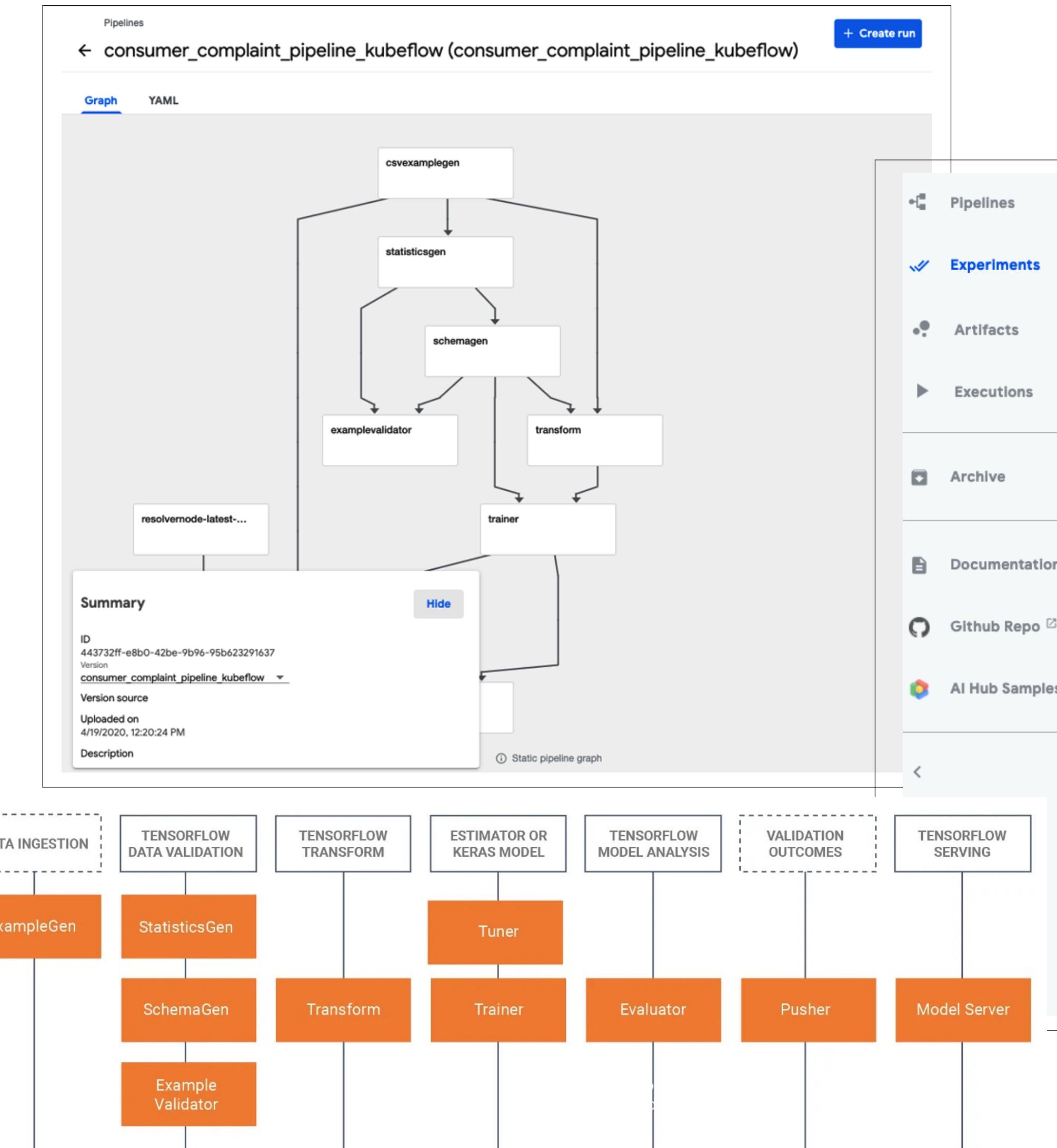


- Plateforme d'apprentissage automatique gratuite et open-source conçue pour permettre l'utilisation de pipelines d'apprentissage automatique afin d'orchestrer des flux de travail compliqués
- Conçu à la base par Google pour déployer in interne des modèles TensorFlow (2018)
- Exécution sur Kubernetes pour fonctionner sur différentes plateformes Cloud (on-premise, GCP, AWZ, Azure) et permettre un déploiement complexe et des mises à l'échelle
- Boîte à outils d'orchestration de pile ML de bout en bout
- Approche par composants pour définir les tâches de la pipeline pour faciliter la réutilisation entre pipelines
- Gestion détaillée d'un projet et suivi et analyse en profondeur
- JupyterHub pour permettre à plusieurs utilisateurs de contribuer à un projet simultanément
- Combinaison avec TensorFlow Extended

**Kubeflow**



# Kubeflow<sup>2</sup>



**Kubeflow**

Select namespace ▾

[Dashboard](#) [Activity](#)

**Quick shortcuts**

- [Upload a pipeline](#) Pipelines
- [View all pipeline runs](#) Pipelines
- [Create a new Notebook server](#) Notebook Servers
- [View Katib Studies](#) Katib
- [View Metadata Artifacts](#) Artifact Store

**Recent Notebooks**

Choose a namespace to see Notebooks

**Recent Pipelines**

- [Sample] Basic - Exit Handler Created 8/12/2019, 2:16:12 PM
- [Sample] Basic - Conditional execution Created 8/12/2019, 2:16:11 PM
- [Sample] Basic - Parallel execution Created 8/12/2019, 2:16:10 PM
- [Sample] Basic - Sequential execution Created 8/12/2019, 2:16:09 PM
- [Sample] ML - TFX - Taxi Tip Prediction Model T... Created 8/12/2019, 2:16:07 PM

**Google Cloud Platform**

- Stackdriver Logging View cluster logs for the past hour
- Project Overview Manage your GCP Project
- Deployment Manager View your deployments
- Kubernetes Engine Administer your GKE clusters

**Documentation**

- Getting Started with Kubeflow Get your machine-learning workflow up and running on Kubeflow
- MiniKF A fast and easy way to deploy Kubeflow locally
- MicroK8s for Kubeflow Quickly get Kubeflow running locally on native

**Job info**

Job name	beamapp-root-0424022141-991143
Job ID	2020-04-23_19_21_48-2536048590988335140
Job type	Batch
Job status	Succeeded
SDK version	Apache Beam Python 3.6 SDK 2.17.0
Region	us-central1
Start time	April 23, 2020 at 7:21:50 PM GMT-7
Elapsed time	29 min 7 sec
Encryption type	Google-managed key

**Resource metrics**

Current vCPUs	1
Total vCPU time	0.649 vCPU hr
Current memory	3.75 GB
Total memory time	2.436 GB hr
Current PD	50 GB
Total PD time	32.479 GB hr
Current SSD PD	0 B
Total SSD PD time	0 GB hr
Total Shuffle data processed	646.16 KB
Billable Shuffle data processed	323.08 KB

**Labels**

**Run detail**

← Start Run detail

Pipeline \* consumer\_complaint\_pipeline\_kubeflow

Pipeline Version consumer\_complaint\_pipeline\_kubeflow

consumer\_complaint\_pipeline\_kubeflow

Privacy • Usage Reporting build version v1beta1

Run name\* Run of consumer\_complaint\_pipeline\_kubeflow (5aaaf5)

Description (optional)

This run will be associated with the following experiment

Experiment\*

**Run Type**

One-off  Recurring

**Run parameters**

Specify parameters required by the pipeline

pipeline-root /tfx-data/output/consumer\_complaint\_pipeline\_kubeflow

**Start** **Cancel**

# Amazon Sage Maker



Amazon  
SageMaker

- Utilisation de blocs-notes Jupyter hébergés
- Outil d'exploration et de visualisation des données d'entraînement stockées dans Amazon S3
- Connection directe aux données de S3 pour les analyser dans les blocs-notes et possibilité d'utiliser AWS Glue pour déplacer les données d'Amazon RDS, Amazon DynamoDB et Amazon Redshift
- Mise à disposition de 10 algorithmes d'apprentissage automatique les plus courants qui ont été préinstallés et optimisés pour offrir des performances jusqu'à 10 fois supérieures à celles que vous trouverez en exécutant ces algorithmes ailleurs.
- Préconfiguration pour TensorFlow et Apache MXNet mais possibilité d'utiliser un framework spécifique
- Console d'entraînement One-click et gestion automatisée de l'infrastructure pour former des modèles à l'échelle du pétaoctet.
- Règlage automatiquement des hyperparamètres du modèle pour obtenir la meilleure précision possible (AutoML)
- Déploiement de modèle sur un cluster d'instances Amazon EC3 avec mise à l'échelle automatique, réparties sur plusieurs zones de disponibilité afin d'offrir à la fois des performances et une disponibilité élevées.
- Fonctionnalités intégrées de test A/B pour faciliter les tests des modèles et à expérimenter différentes version

# Amazon Sage Maker

Amazon SageMaker Studio File Edit View Run Kernel Git Tabs Settings Help

xgboost\_customer\_churn.ipynb X

Markdown git conda\_amazonei\_mxnet\_p27

- Have the predictor variable in the first column
- Not have a header row

But first, let's convert our categorical features into numeric features.

```
[ ]: model_data = pd.get_dummies(churn)
model_data = pd.concat([model_data['Churn?_True.'], model_data.drop(['Churn?_True.'])], axis=1)
```

And now let's split the data into training, validation, and test sets. This will help prevent us from overfitting the model, and allow us to test the models accuracy on data it hasn't already seen.

```
[ ]: train_data, validation_data, test_data = np.split(model_data.sample(frac=1), 3)
train_data.to_csv('train.csv', header=False, index=False)
validation_data.to_csv('validation.csv', header=False, index=False)
```

Now we'll upload these files to S3.

```
[ ]: boto3.Session().resource('s3').Bucket(bucket).Object(os.path.join(prefix, 'train.csv')).upload_file('train.csv')
boto3.Session().resource('s3').Bucket(bucket).Object(os.path.join(prefix, 'validation.csv')).upload_file('validation.csv')
```

Trial Component Chart

The chart displays the 'train:loss:last' metric over time (0, 1, 2) for four trials. The y-axis ranges from 0.0 to 0.4. Trial 1 (blue) starts at ~0.35 and drops to ~0.1. Trial 2 (green) starts at ~0.38 and drops to ~0.15. Trial 3 (red) starts at ~0.25 and drops to ~0.05. Trial 4 (orange) starts at ~0.15 and drops to ~0.05.

Amazon SageMaker Studio Components and registries Select the component or registry to view.

Endpoints 1 row selected 0/20 filters Search column name to start

Name	Endpoint status
seanmign-test-staging	InService
DEMO-xgb-churn-model-monito...	InService
DEMO-xgb-churn-model-monito...	InService
DEMO-xgb-churn-model-monito...	InService
DEMO-xgb-churn-model-quality...	InService
xai-analyzer-german-xgb	InService
xai-analyzer-german-xgb-enabl...	InService
xai-analyzer-german-xgb-2	InService
xai-analyzer-german-xgb2020-1...	InService
UffNetModel1-training	InService

TRIAL COMPONENT 10 rows selected

Status
Completed

Amazon SageMaker Studio

File Edit View Run Kernel Git Tabs Settings Help

Components and registries

Select the component or registry to view.

Pipelines

Search column name to start

Name: my-demo Clear all

Name Last modified

my-demo-project-p-v04giyx... 2 minutes ago

End of the list

Graph Parameters Settings

Search for step...

ProjectAbaloneProcess

ProjectAbaloneTrain

ProjectAbaloneEval

ProjectAbaloneMSECond

true

11/24/2020, 9:34 AM 10m55s

Status Started time Elapsed time

Git Tabs Settings Help

SageMaker-ModelMonitoring x xai-analyzer-german-xgb-ena x

half a minute ago

MODEL MONITORING

Endpoint: xai-analyzer-german-xgb-enable-data-capture

Data quality Model explainability Bias drift AWS settings

AMAZON SAGEMAKER BIAS DRIFT MONITORING

Detect bias drift and other issues that affect models in production and receive alerts you so you can take corrective action.

MONITORING JOB HISTORY

Monitoring status	Monitoring job name	Monitoring schedule name	Created
No issues	model-bias-monitoring-2020111...	mm-v2-model-bias-schedule-202...	46 minutes ago
Issue found	model-bias-monitoring-2020111...	mm-v2-model-bias-schedule-b-2...	48 minutes ago
No issues	model-bias-monitoring-2020111...	mm-v2-model-bias-schedule-202...	2 hours ago
Issue found	model-bias-monitoring-2020111...	mm-v2-model-bias-schedule-b-2...	2 hours ago
Issue found	model-bias-monitoring-2020111...	mm-v2-model-bias-schedule-b-2...	3 hours ago
No issues	model-bias-monitoring-2020111...	mm-v2-model-bias-schedule-202...	3 hours ago
Failed		mm-v2-model-bias-schedule-b-2...	4 hours ago
Failed		mm-v2-model-bias-schedule-202...	4 hours ago
Failed		mm-v2-model-bias-schedule-b-2...	5 hours ago

Add chart

1 CHARTS

bias\_metric\_CDDL: Average

Bias metric value

Date and time

source Observed Threshold

BIAS DRIFT CHART PROPERTIES

Label: Class1Good2Bad

Label value: 1

Timeline

1 week 1 day 12 hours 6 hours 1 hour

Facet

ForeignWorker

Facet value

1

Monitoring schedule

mm-v2-model-bias-schedule-b-2020-10-25-20-1...

Bias metric

bias\_metric\_CDDL

Threshold

-1