GenAI : Industrialiser sans compromettre

Guide du Build vs Buy

François PAUPIER - 22 janvier 2024

Identifier & actionner les leviers pour accélérer sa delivery

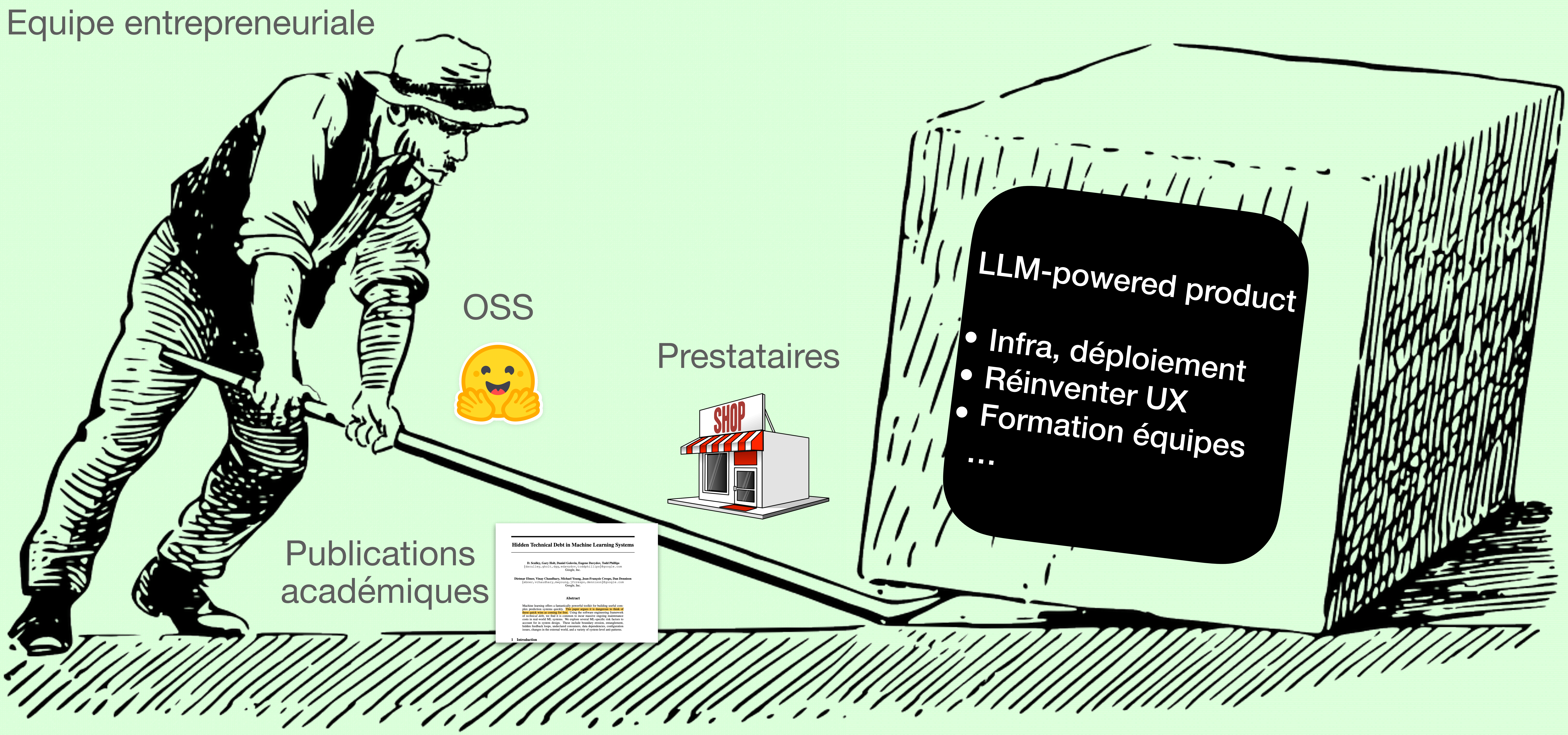


image source: OpenClipart-Vectors, Pixabay

Agenda

1. **Top #1 erreur observée et comment l'éviter**

2. Build vs Buy - clés pour assurer sa différenciation

3. Anti-patterns récurrents et comment s'en prémunir

1. Top #1 erreur observée et comment l'éviter

Top #1 erreur ⚠

Feuille de route GenAI pilotée par les trending repos GitHub

1. Top #1 erreur observée et comment l'éviter

When "Agents" replaces RAG as GitHub trending topic

1. Top #1 erreur observée et comment l'éviter

Identifier ses facteurs de différenciation

A - Permettre un gain de productivité x10
sur sa verticale métier

B - Débloquer de nouveaux usages
sur sa verticale métier

MINES PARIS | PSL
Switching perspectives on generative ar...
minesparis-psl.hal.science

Repenser la place de l'utilisateur dans la boucle décision - action

1. Top #1 erreur observée et comment l’éviter

Diagnostic de son org pour identifier les différenciants atteignables

Un mot sur la HR

Formation / Recrutement / Prestataires

1 - Lancer le produit

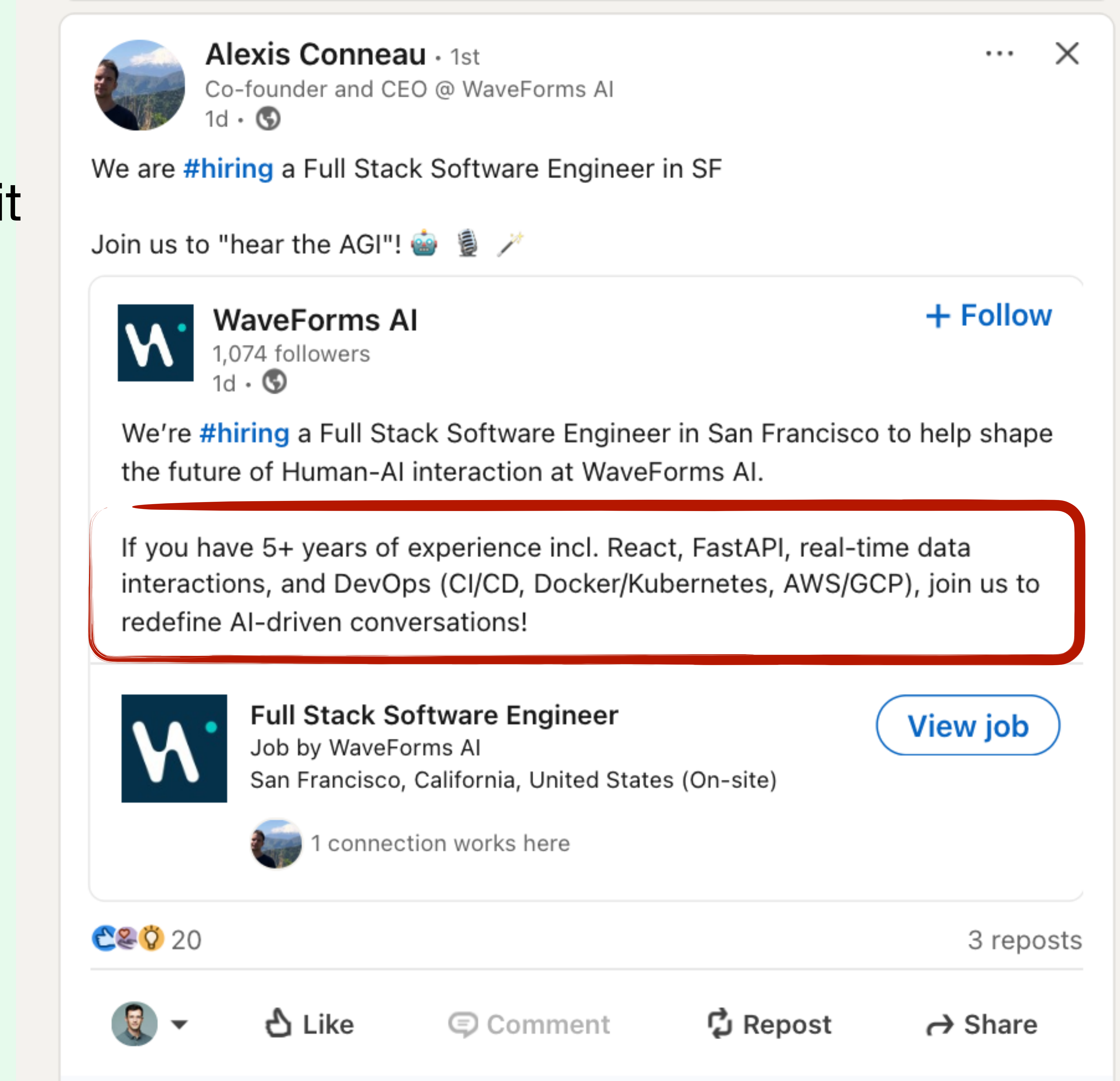
Full stack
DevOps
AI engineer (= full stack dev qui sait consommer des API GenAI)

2 - Ancrer ses fondations data

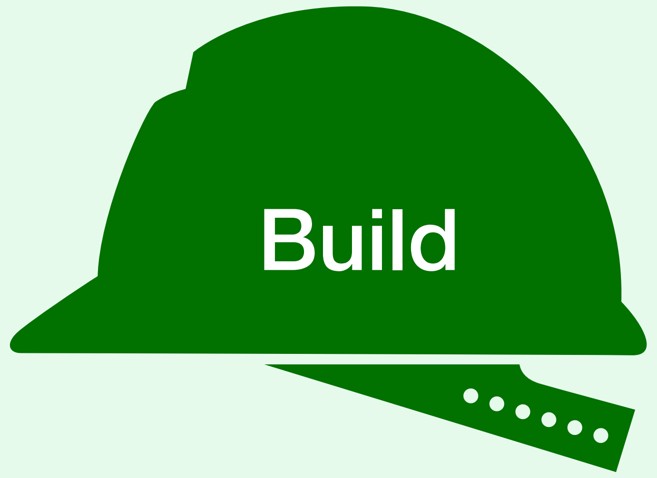
Data Engineer
ML Engineer

3 - Optimiser ses algos

ML Engineer
Data Scientist

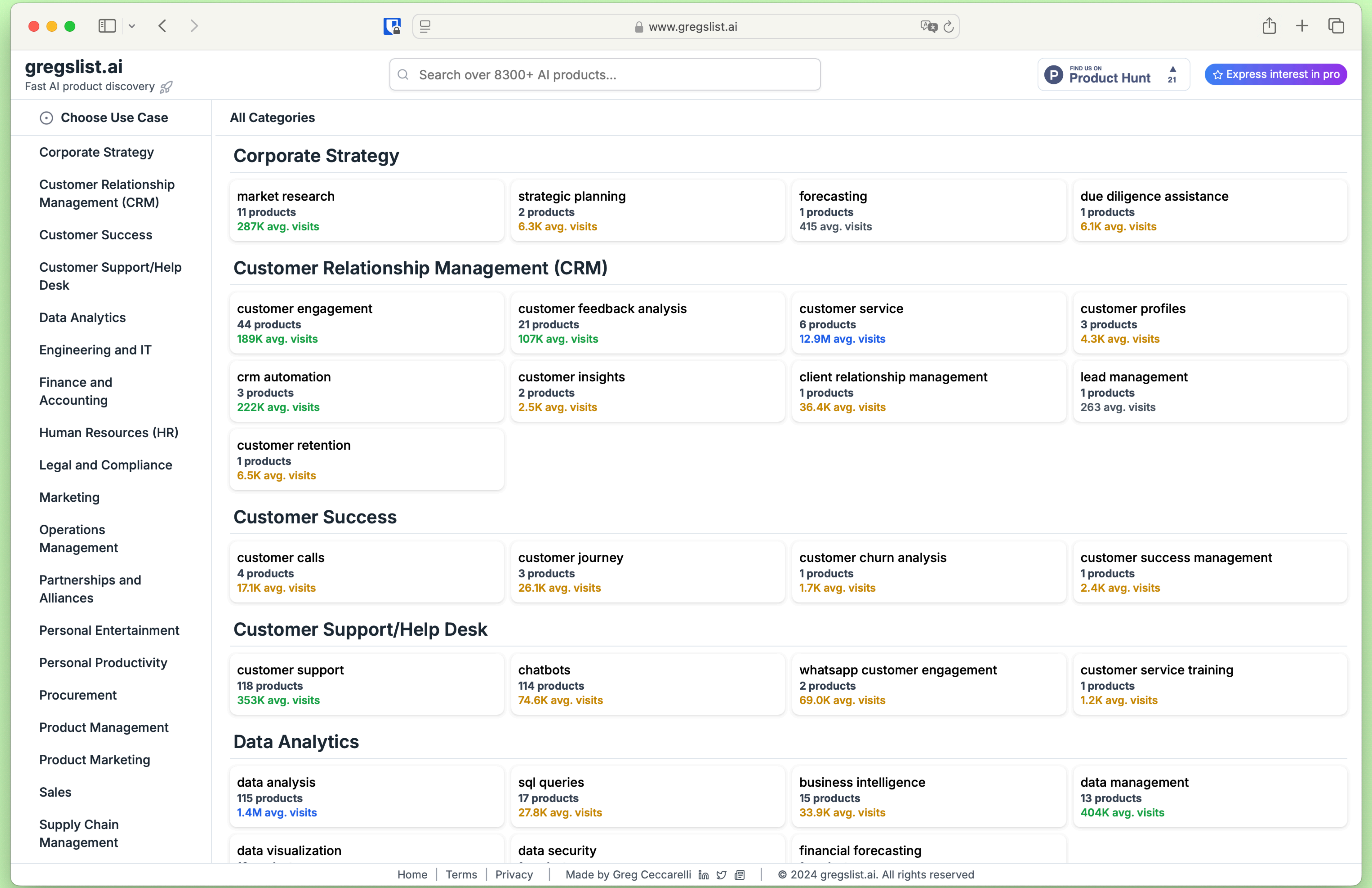


2. Build vs Buy - clés pour assurer sa différenciation



- **Brique critique** de votre chaîne de valeur
- L’investissement (temps/€) crée un actif **durable** et **réutilisable**
- Créer/entretien votre *unfair advantage*

- Les solutions existantes sont **matures** et bien **maintenues**
- **Effet d'échelle** bénéficiant aux géants du web / prestataires experts
- Le composant est une **commodité** - transverse à plusieurs industries



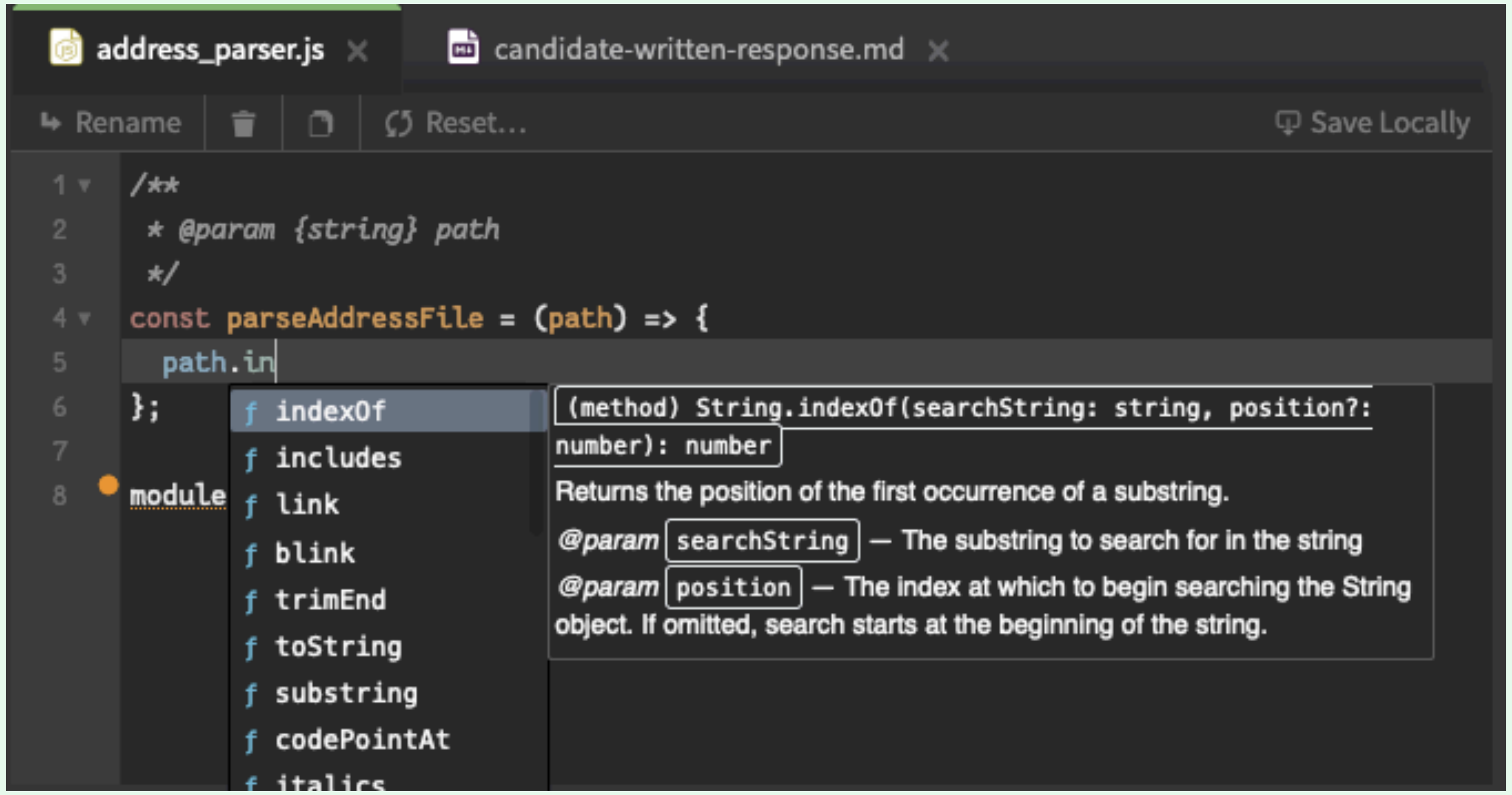
Ma feature IA est-elle une commodité ?

@fpaupier

Source: https://www.gregslist.ai/

Exemples de *fausse bonne idée* de build nº1

Créer son code assistant



Hébergement du LLM spécialisé code

Intégration dans toutes les features IDE au de la de la code completion

Quel ROI vs un abonnement 10$/month/user ?..

Exemples de *fausse bonne idée* de build nº2

Créer son RAG pour la doc

Chaine de RAG sur sa docs

customer facing

sur sa doc usage interne

Maintenance des connecteurs aux sources de données - **chronophage**

Gestion du besoin d'en connaitre (compliance, légal)
Mécanisme d'**Access Control non trivial**

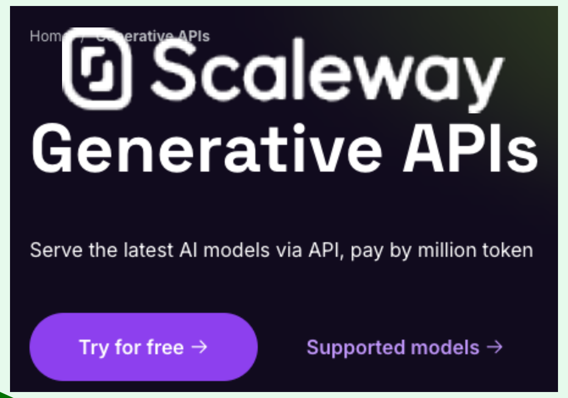
Mise à jour de la logique d'indexing / embedding
Forte expertise ML Eng

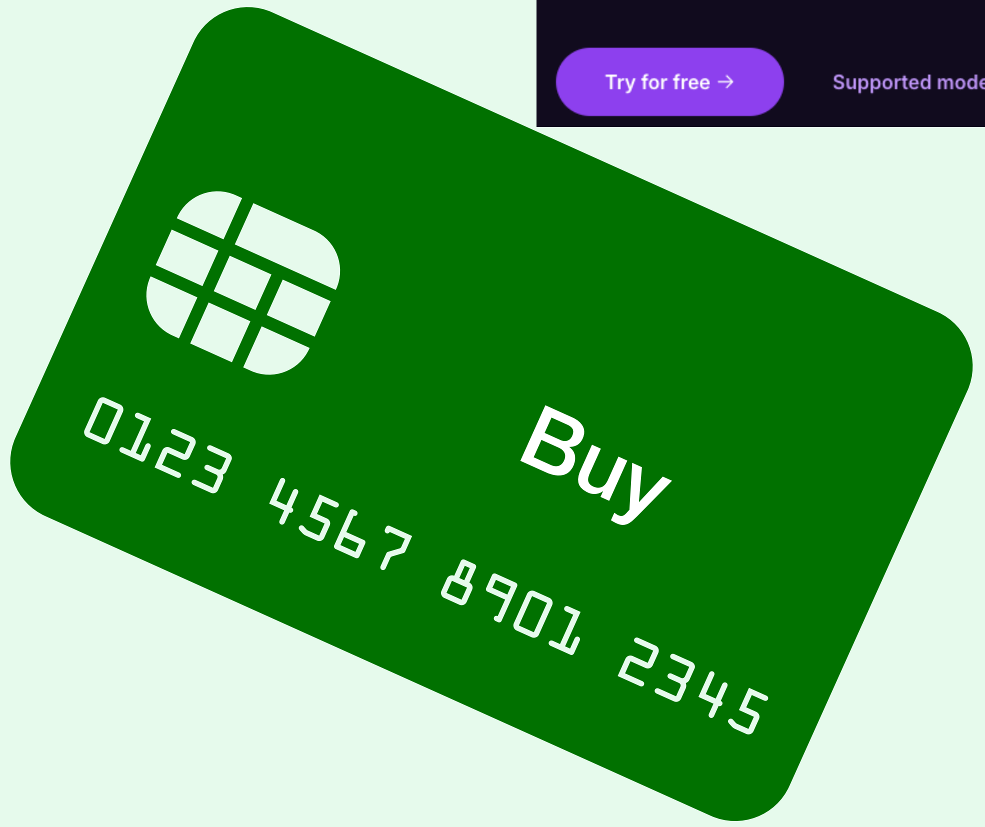
Mise à jour de la chaine de RAG
Full time Job

Faible niveau de différentiation produit

Exemples de *fausse bonne idée* de build nº3

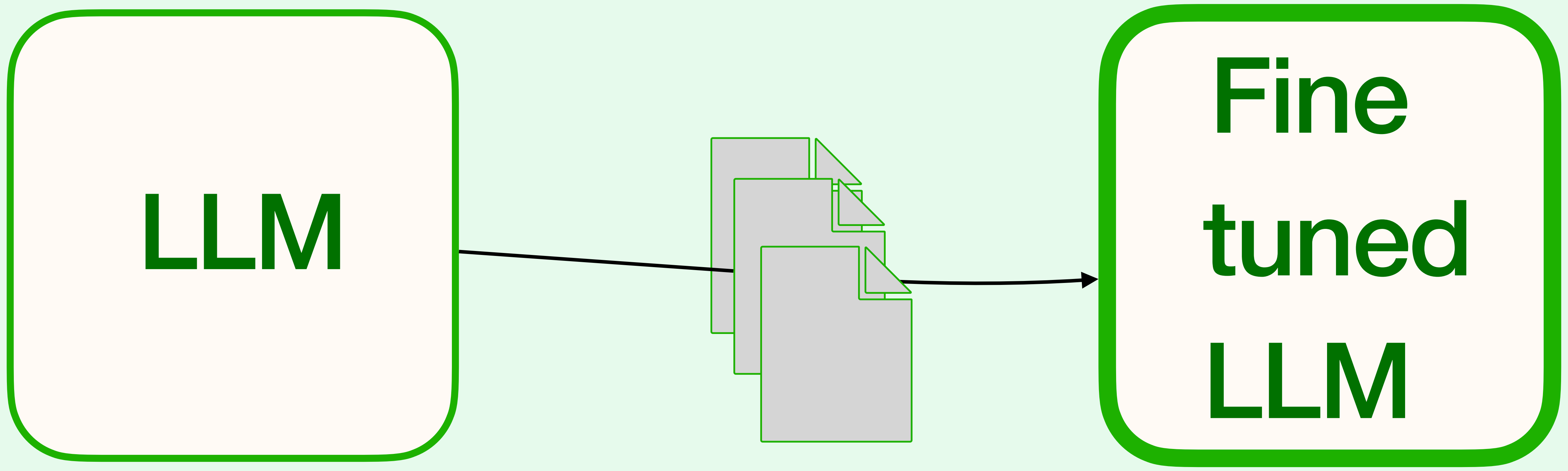
Claude 3.5 Sonnet





Fine tuner un LLM pour son cas d'usage

OpenAI developer platform



Dataset spécifiques sur votre tâche business

Nécessite jeu d'évaluation souvent, au mieux incomplet

Nouveau SOTA tous les mois

Risque de perte en qualité de généralisation
Forte expertise ML Eng / DS

Nécessite dataset d'entrainement de qualité

Haut risque / faible récompense
Performance du LLM n'est plus le premier bottleneck

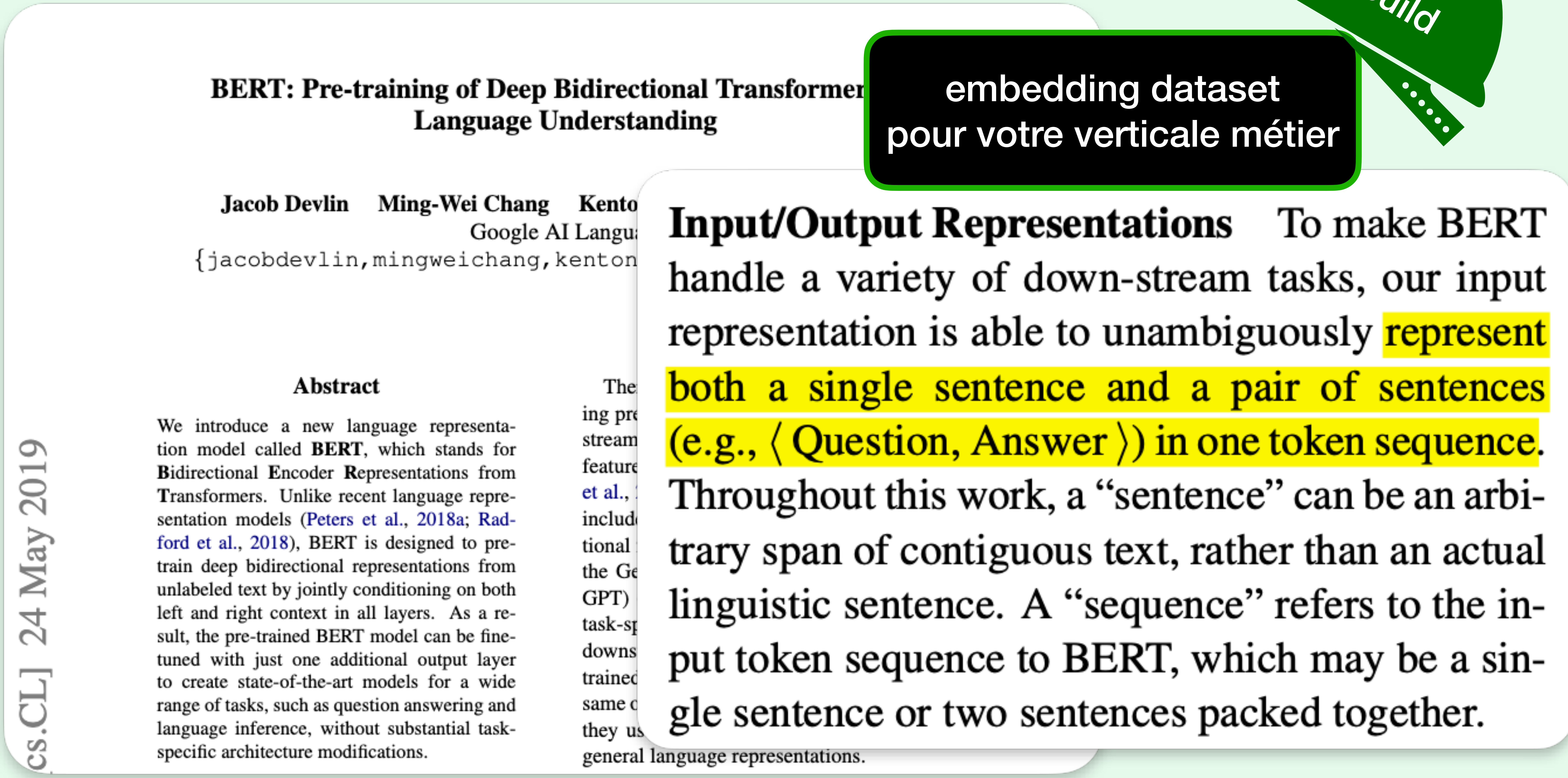
@fpaupier

Que construire en interne?   Exemple #1 NLP

Regarder les invariants et miser sur des assets de temps long

2018 - BERT

2024 - BGE



BERT: Pre-training of Deep Bidirectional Transformer
Language Understanding

Jacob Devlin   Ming-Wei Chang   Kento
Google AI Langu
{jacobdevlin,mingweichang,kenton

arXiv [cs.CL] 24 May 2019

Abstract

We introduce a new language representation model called **BERT**, which stands for **B**idirectional **E**ncoder **R**epresentations from **T**ransformers. Unlike recent language representation models (Peters et al., 2018a; Radford et al., 2018), BERT is designed to pretrain deep bidirectional representations from unlabeled text by jointly conditioning on both left and right context in all layers. As a result, the pre-trained BERT model can be fine-tuned with just one additional output layer to create state-of-the-art models for a wide range of tasks, such as question answering and language inference, without substantial task-specific architecture modifications.

general language representations.

Input/Output Representations To make BERT handle a variety of down-stream tasks, our input representation is able to unambiguously represent both a single sentence and a pair of sentences (e.g., ⟨ Question, Answer ⟩) in one token sequence. Throughout this work, a "sentence" can be an arbitrary span of contiguous text, rather than an actual linguistic sentence. A "sequence" refers to the input token sequence to BERT, which may be a single sentence or two sentences packed together.

embedding dataset
pour votre verticale métier

Build

Source: {Devlin et al., 2018}

Que construire en interne?   Exemple #2 Computer Vision

Maitriser sa stack monitoring / evaluation

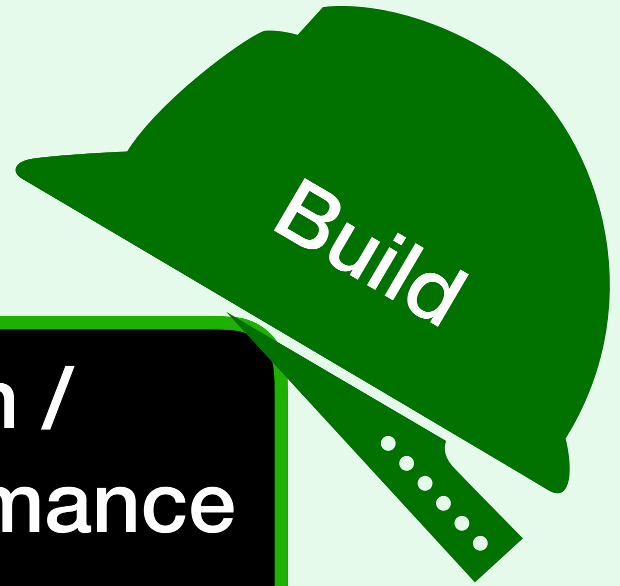
PRELIGENS

- Traitement d'images satellite pour détection & suivi d'activité

Source: Safran AI (ex Preligens)

Que construire en interne? Exemple #2 Computer Vision

Maitriser sa stack monitoring / evaluation



Stack évaluation / observabilité performance modèles

Our strategy :

- Focus on research and development : A dedicated R&D team to test and integrate state of the art research into our models, in test but also in production.
- Create a unique framework : our AI factory. A toolbox for our AI teams, helping them to create in a short period of time high accuracy and reliable detectors, to industrialize AI algorithms creation.
- Dedicate a team to this framework : our AI engineering team. A third of our AI engineers are working on the framework. This is a multidisciplinary team, from Machine Learning engineers, to computer vision researchers, architects, fullstack developers and devOps.

VALOHAI

Prestataires

Mise en place stratégie+stack d'évaluation

Source: Safran AI (ex Preligens), AI Factory blog post, Medium.com

Que construire en interne? Exemple #3 Agents

Encapsuler une expertise métier niche et combiner votre ETL existant avec des modules LLM

> AI Agents are **programs where LLM outputs control the workflow.**

Any system leveraging LLMs will integrate the LLM outputs into code. The influence of the LLM's input on the code workflow is the level of agency of LLMs in the system.

Note that with this definition, "agent" is not a discrete, 0 or 1 definition: instead, "agency" evolves on a continuous spectrum, as you give more or less power to the LLM on your workflow.

See in the table below how agency can vary across systems:

Agency Level	Description	How that's called	Example Pattern
☆☆☆	LLM output has no impact on program flow	Simple Processor	`process_llm_output(llm_response)`
★☆☆	LLM output determines an if/else switch	Router	`if llm_decision(): path_a() else: path_b()`
★★☆	LLM output determines function execution	Tool Caller	`run_function(llm_chosen_tool, llm_chosen_args)`
★★★	LLM output controls iteration and program continuation	Multi-step Agent	`while llm_should_continue(): execute_next_step()`
★★★	One agentic workflow can start another agentic workflow	Multi-Agent	`if llm_trigger(): execute_agent()`

Source: Hugging Face, Introduction to Agent, retrieved Jan 13, 2024

Que construire en interne? Exemple #3 Agents

Faire un virement depuis une app mobile

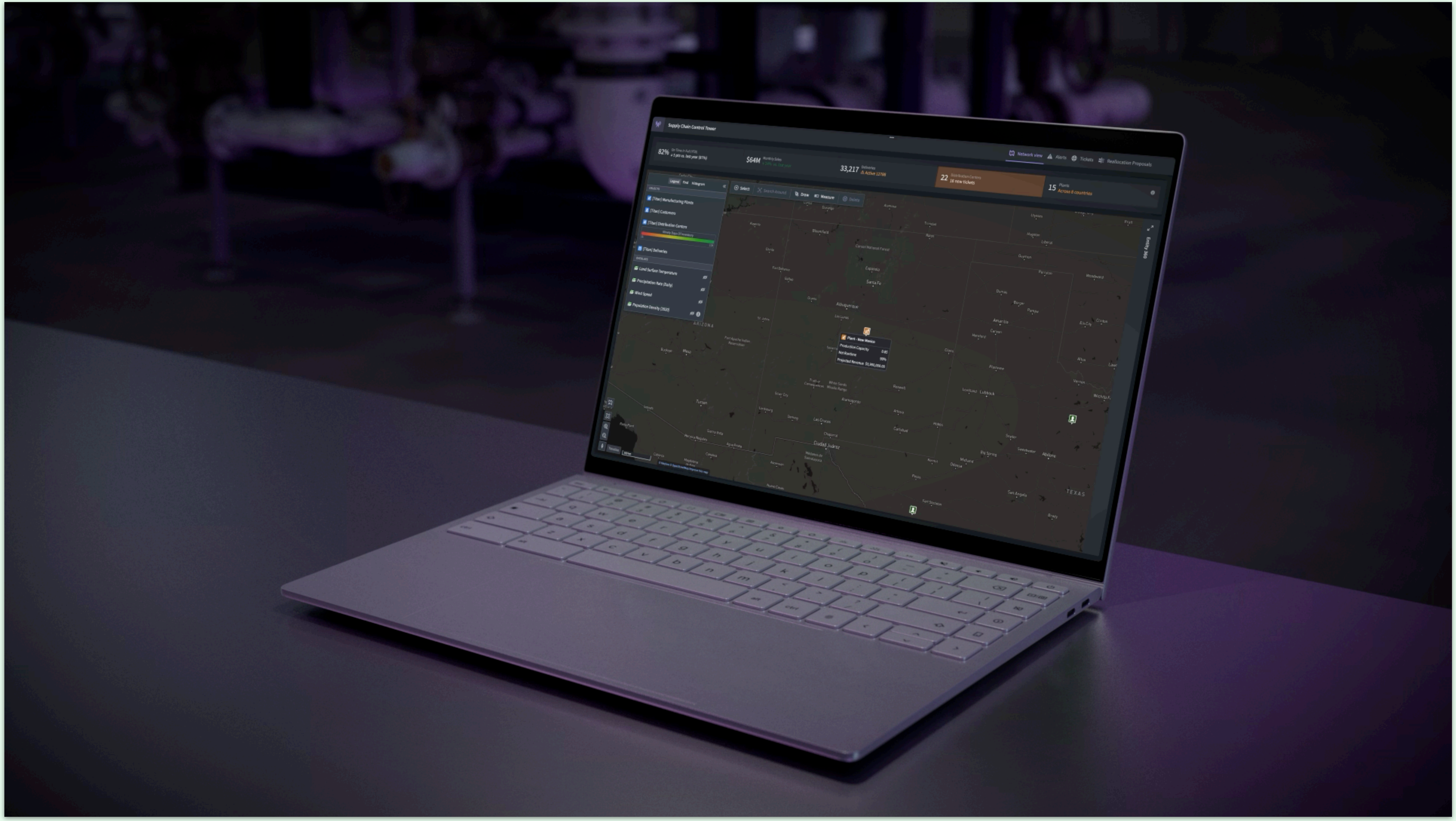
Pas besoin d'agent

Proposer des suggestions de scénarios dans un outil d'aide à la gestion de crise

- Catégoriser l'événement
- Prise en compte des événements et résolutions historiques
- Prise en compte des Standard operating procedures (SOPs)

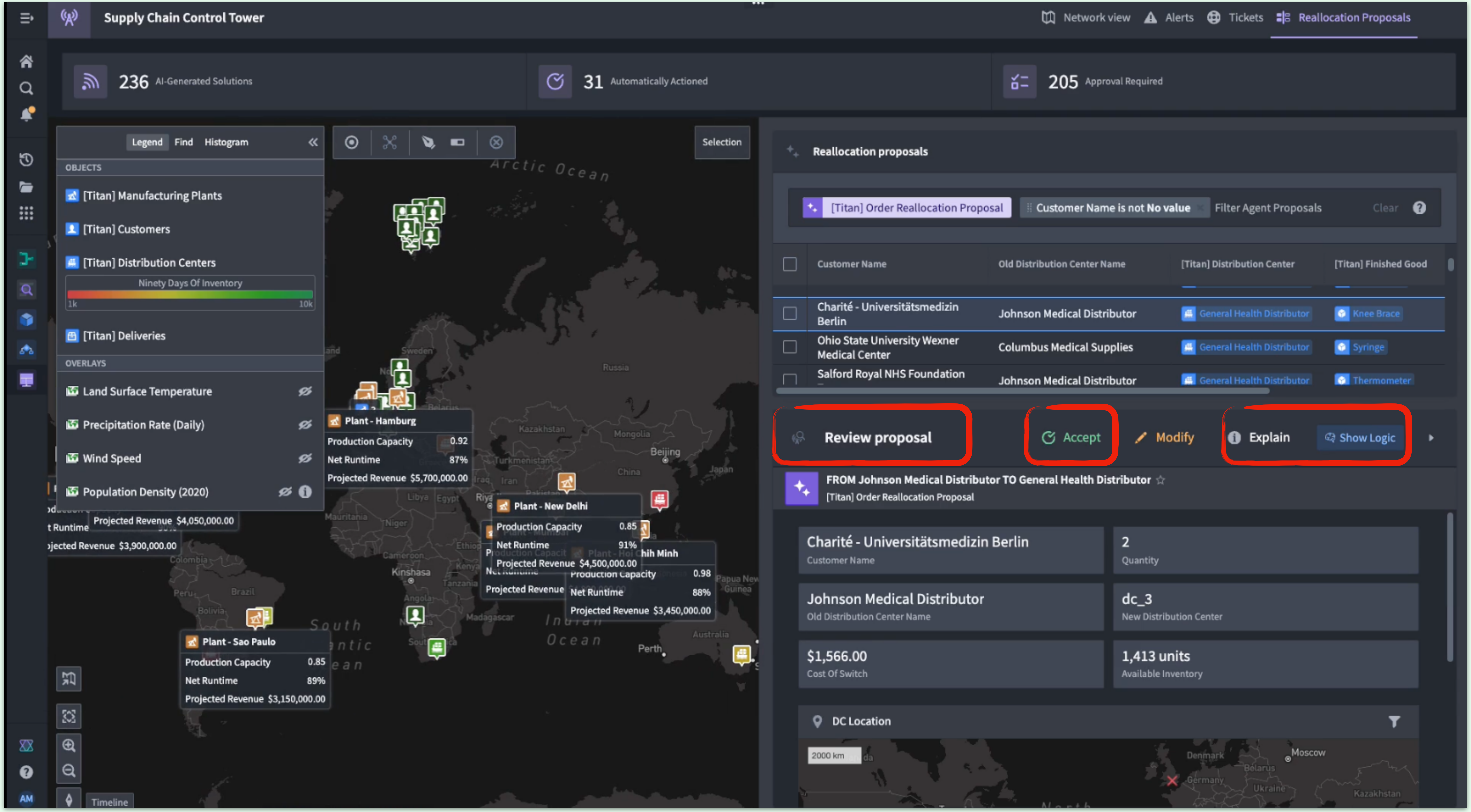
Logique métier + LLM sur une verticale précise
"multi step agent"

Que construire en interne?   Exemple #3 Agents



Source: Palantir, Platform AIP, retrieved Jan 13 2025

Que construire en interne? Exemple #3 Agents



Source: Palantir, Platform AIP, retrieved Jan 13 2025

3. Anti-patterns récurrents et comment s'en prémunir

Hidden Technical Debt in Machine Learning Systems

D. Sculley, Gary Holt, Daniel Golovin, Eugene Davydov, Todd Phillips
{dsculley,gholt,dgg,edavydov,toddphillips}@google.com
Google, Inc.

Dietmar Ebner, Vinay Chaudhary, Michael Young, Jean-François Crespo, Dan Dennison
{ebner,vchaudhary,mwyoung,jfcrespo,dennison}@google.com
Google, Inc.

Abstract

Machine learning offers a fantastically powerful toolkit for building useful complex prediction systems quickly. This paper argues it is dangerous to think of these quick wins as coming for free. Using the software engineering framework of *technical debt*, we find it is common to incur massive ongoing maintenance costs in real-world ML systems. We explore several ML-specific risk factors to account for in system design. These include boundary erosion, entanglement, hidden feedback loops, undeclared consumers, data dependencies, configuration issues, changes in the external world, and a variety of system-level anti-patterns.

Date de 2015

Etonnement d'actualité à l'ère des LLMs!

Source: {D. Sculley, 2015}

Zoom sur trois anti patterns

Configuration debt

Glue code

Pipeline Jungles

Anti pattern - Glue code

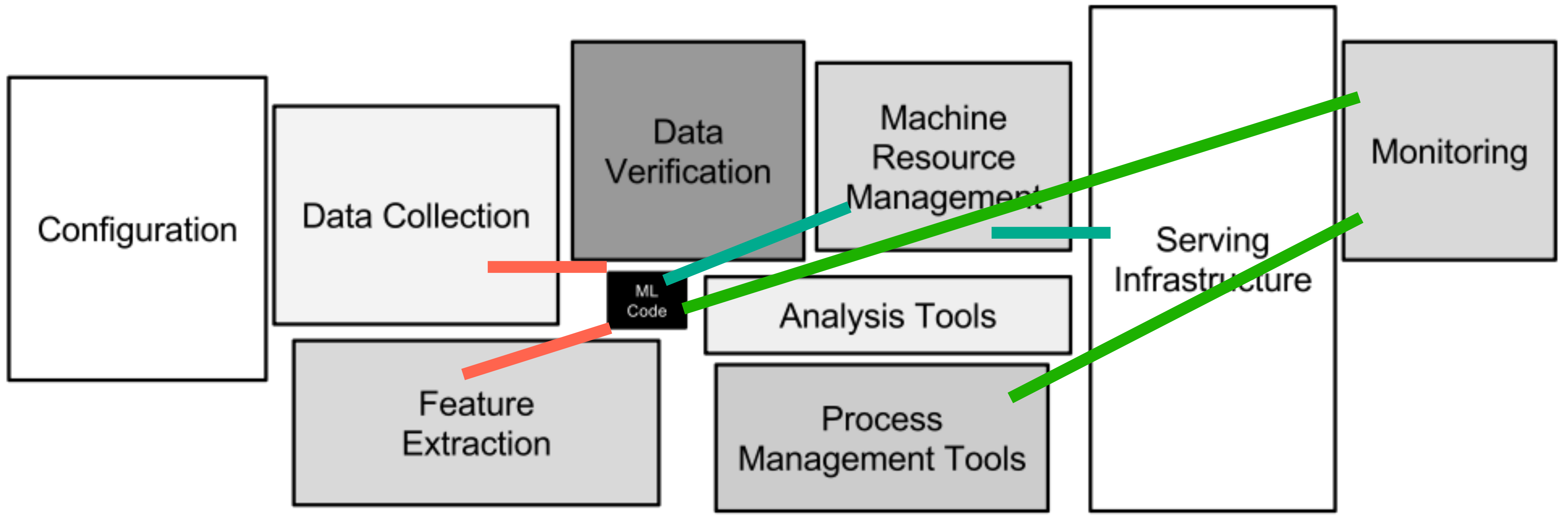


Figure 1: Only a small fraction of real-world ML systems is composed of the ML code, as shown by the small black box in the middle. The required surrounding infrastructure is vast and complex.

> *Because a mature system might end up being (at most) 5% machine learning code and (at least) 95% glue code, **it may be less costly to create a clean native solution** rather than re-use a generic package.*

Usage à surveiller

LangChain

LlamaIndex

Painkillers

Domain-model clairement défini
Utiliser pattern *adapter/facade* pour
ne pas dépendre de l'implem. d'un tiers ex OpenAI API
Compatible LLM inference server

Source: {D. Sculley, 2015}

Anti pattern - Configuration debt

image source: alexanderstein Pixabay

> *In a mature system which is being actively developed, the **number of lines of configuration can far exceed** the number of lines of the traditional code. Each configuration line has a potential for mistakes.*

Point d'attention

Config. clairsemé partout dans l'app

Pas de single source of truth

Difficulté de répliquer un déploiement

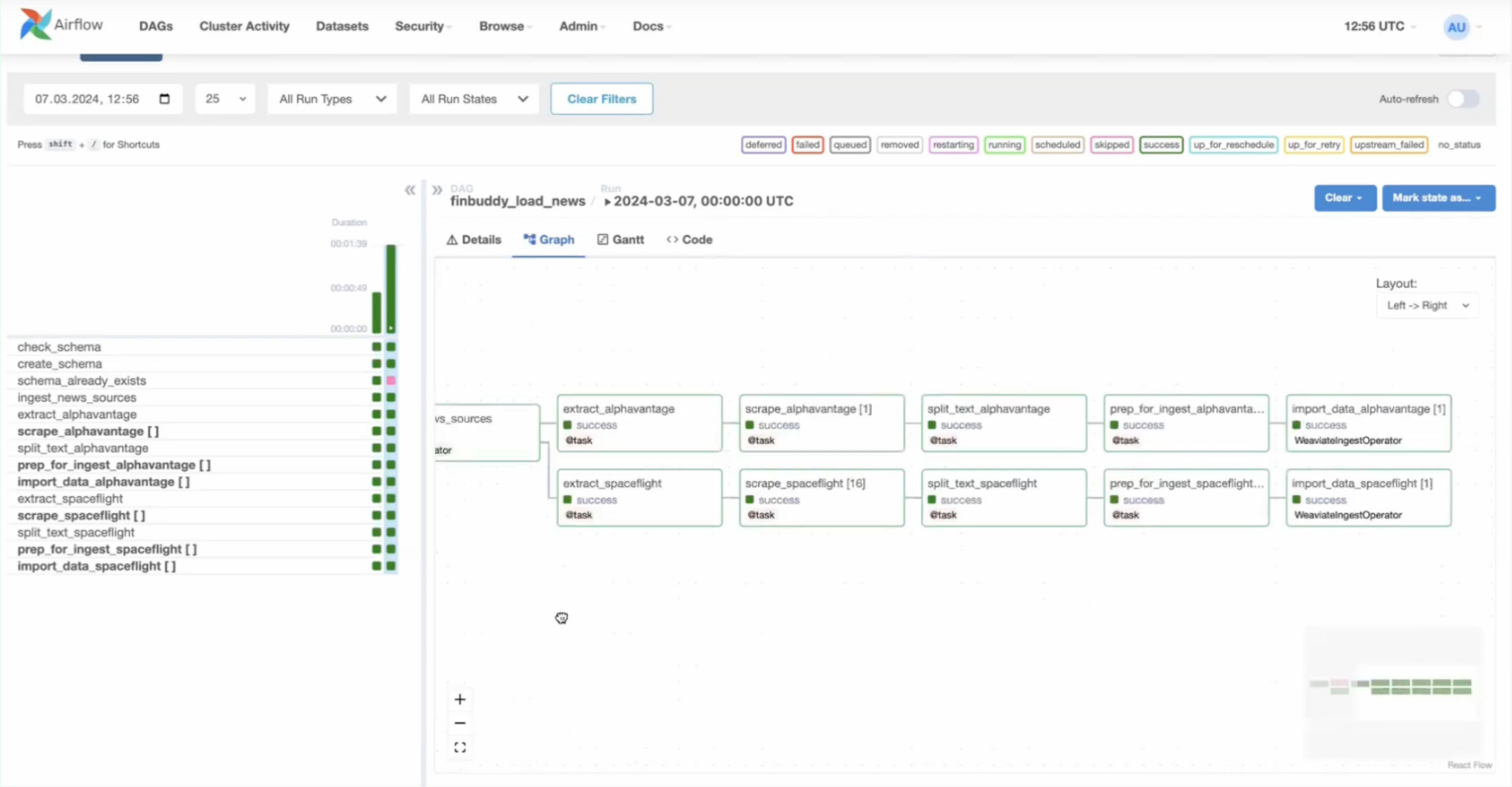
Painkillers

Config. Sous VCS

Verifier la conf "critical path" dans la CI

Source: {D. Sculley, 2015}

Anti pattern - Pipeline Jungles



Source: Apache Airflow ML Ops,

> *Glue code and pipeline jungles are symptomatic of integration issues that may have a root cause in overly separated “research” and “engineering” roles*

Point d'attention

Lancer le pipeline de pre-processing donne la boule au ventre

Pas d’erreur state propre - données corrompue

Painkillers

Définir error state et instruction de retry

Composant du pipeline +modulaire

Source: {D. Sculley, 2015}

Comment un nouveau modèle impacte les performances de la prod ?

Peut-on tester une nouvelle approche à l'échelle rapidement ?

Questions utiles

A quelle vitesse un nouveau membre de l'équipe peut être onboardé sur la stack?

Comment savoir le data lineage pour une sortie du système ?

Appendix

PDF slides available on

fpaupier.fr/202501_productionize_genai

Bibliographie et sources

- **First hands experience building & deploying LLMs in small & large orgs.**

- D. Sculley, G. H., Daniel Golovin, Eugene Davydov, Todd Phillips. (2015). Hidden Technical Debt in Machine Learning Systems. NIPS.
- Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2018). BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv, 1810.04805v2. http://arxiv.org/abs/1810.04805v2
- Antoine Bordas, Pascal Le Masson, Benoit Weil. Switching perspectives on generative artificial intelligence: a design view for humans-generative AI co-creativity. R&D Management Conference 2024, Jun 2024, Stockholm, Sweden. hal-04520521
- Asano, Y. (2024). Parameter-Efficient Fine Tuning. Retrieved Jan 13 2025, https://uvafomo.github.io/. University of Amsterdam
- Paupier, F. (2024, Sep 18). REX – GenAI in prod : Practical tech & product insight for delivering value with RAG
- Shreya Shankar, Data Flywheels for LLM Applications, Jul 1, 2024
- Hamel Husain, AI Essentials For Tech Executives, https://ai-execs.com/, 3 January 2025
- US DoD Directive 3000.09 Autonomy in weapon systems, January 25, 2023
- Lt Col Joseph O. Chapa, USAF, Please Stop Saying “Human-In-The-Loop”, September 3, 2024
- Paupier F. (2021, Oct 19). Le ML Ops pour accélérer l’innovation, de la R&D à la production, Interview Preligens - Post Mortem Podcast
- Qonto, How Qonto Achieved 70% Faster Localization and Streamlined Operations with AI Assistants, 19 Sep 2024
- Hugging Face, Introduction to Agent, retrieved Jan 13 2025
- Palantir, Platform AIP, retrieved Jan 13 2025