Techstrong | Special Report

PART OF Futurum MEDIA

OCTOBER 2026

OPEN-WEIGHT AI

CAPABILITY · INFRASTRUCTURE · ECONOMICS

Open Weights, Real Costs

How open-weight AI compares with frontier models, what it takes to run it, and when the economics work

BY ALAN SHIMEL



Competitive at which tasks?
Where open-weight models stand against the proprietary frontier

Four purchases that should not be confused
SaaS, APIs, managed inference and self-hosting

Three workloads, three economic decisions
From a 25-person team to 70 billion tokens a month

71.5%
Open-weight share of tokens on Vercel's AI Gateway in September

23%
Their share of spending on the same platform

4–7 mo.
Open-weight lag behind the frontier in UK AISI cyber tests

Contents

Introduction 3

PART I · HOW FAR HAVE OPEN-WEIGHT MODELS COME? 4

01 What enterprises actually get when they choose open 5

02 Competitive at which tasks? 6

SIDEBAR *Five questions to ask before calling a model open source* 7

03 Cybersecurity deserves three separate evaluations 8

SIDEBAR *What the benchmarks answer* 9

04 Adoption is becoming a portfolio decision 11

05 Sovereignty requires a dependency inventory 11

SIDEBAR *A practical sovereignty checklist* 12

PART II · WHAT IT COSTS TO PUT THE MODEL TO WORK 13

06 Four purchases that should not be confused 14

07 What it takes to host a model 15

08 Three workloads, three economic decisions 16

SIDEBAR *Fitting a model in memory is only the beginning* 17

SIDEBAR *An illustrative monthly budget that exposes the assumptions* 19

09 The variables that determine the result 20

10 What customer experience tells us 21

SIDEBAR *A worksheet for comparing your own workload* 22

Choosing the operating model you can defend 24

Methodology, limitations and disclosure 25

Sources 25

BY THE NUMBERS

41.5%
of 200 Futurum respondents use open-weight models in production

34%
of AI tokens carried by open weights, as estimated by 83 production users

62%
of open-weight pilots held up by security and compliance reviews

8×B200
validated vLLM deployment for Mistral Large 4 in NVFP4

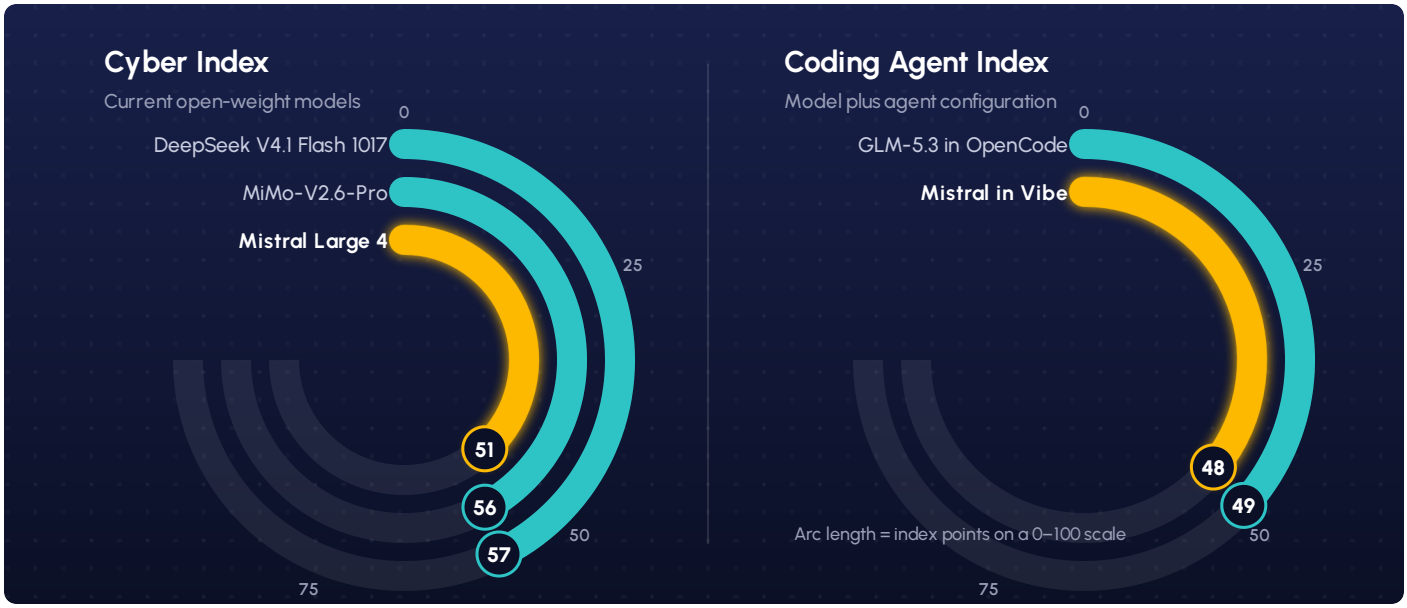
$0.62
per accepted document in the report's illustrative budget

Sources: Futurum; vLLM; report calculations. See Sources.

EXHIBIT 1

Mistral's open-weight case is strong but not a sweep

Artificial Analysis indexes for current open-weight models and configurations, October 29, 2026



Source: Artificial Analysis, Mistral Large 4 release evaluation. Scores apply to specific configurations. [2]

INTRODUCTION

How open-weight AI compares with frontier models, what it takes to run it, and when the economics work

BY ALAN SHIMEL

Mistral chose a memorable name for a very large model. Le Chonk, formally Mistral Large 4, arrived with an equally large claim: the French company's refined open-weight offering could match or surpass leading Chinese models in coding and cybersecurity. WIRED's October 29 coverage put that claim squarely in the competition over who supplies the AI that businesses and governments will depend on. [1] For enterprise buyers, however, a national flag and a benchmark headline are only the beginning of the evaluation.

The independent results give Mistral something substantial to sell, although they do not establish the sweeping victory suggested by the launch pitch. Artificial Analysis placed Large 4 third among current open-weight models on its Cyber Index: 51, behind DeepSeek V4.1 Flash 1017 at 57 and MiMo-V2.6-Pro at 56. On its Coding Agent Index, Mistral in Vibe scored 48, close to GLM-5.3 in OpenCode at 49. Those are specific results for specific configurations, not interchangeable statements about every task the underlying models can perform. [2]

Enterprises should welcome the competition. A credible alternative can expand deployment choices, improve negotiating leverage and make capabilities available to organizations that do not want every request passing through a proprietary model provider. But an enterprise still has to decide whether that alternative completes its work reliably, fits its security requirements and costs less once somebody is responsible for running it. A downloadable model can be an important asset without being a complete product or an economical deployment.

That is the decision this report examines. The first part assesses capability, adoption and control. The second examines the application and infrastructure choices that determine cost. The evidence supports a workload-specific approach: include open weights in serious evaluations, retain proprietary services where their capabilities or convenience justify the expense, and require a concrete operational case before taking on self-hosting. The burden of proof belongs equally to the vendor selling a premium subscription and the team promising savings from operating its own model.



PART I

How Far Have *Open-Weight* Models Come?

01 What enterprises actually get when they choose open

02 Competitive at which tasks?

03 Cybersecurity deserves three separate evaluations

04 Adoption is becoming a portfolio decision

05 Sovereignty requires a dependency inventory

EXHIBIT 2

Selected model access snapshot

Status and license notes at the October 29 cutoff; procurement unit is the exact release and its license

Mistral Large 4, 1T-A52B

DOWNLOADABLE WEIGHTS

Mistral Ecosystem Open-Weight License; examine model-as-a-service provisions

DeepSeek V4.1 Flash 1017

DOWNLOADABLE WEIGHTS

MIT stated for the repository and weights

GLM-5.3

DOWNLOADABLE WEIGHTS

Custom GLM license; qualifying large model-as-a-service businesses face a security-review requirement

MiMo-V2.6-Pro-RL

DOWNLOADABLE WEIGHTS

MIT metadata; this checkpoint should not automatically be equated with every benchmark endpoint labeled MiMo-V2.6-Pro

Qwen3.5-9B

DOWNLOADABLE WEIGHTS

Apache-2.0; a smaller-model example, not a claim of frontier parity

Kimi K3.1 Pro

HOSTED ACCESS; WEIGHTS ANNOUNCED

Vendor says full weights are due November 4; not counted here as a currently downloadable release

MIT or Apache-2.0 stated | Custom license with conditions | Weights announced, not yet downloadable

Sources: official repositories, licenses and release documentation. This is a selected access comparison, not a comprehensive leaderboard. [4, 5, 8, 6, 7, 9, 10, 11]

01

What enterprises actually get when they choose open

Open-weight describes access to a model's learned parameters. Frontier describes capability. They are different dimensions, and an open-weight model can be at or near the frontier on a particular task. Throughout this report, the comparison is with proprietary frontier models where that is the intended meaning. Treating all open models as inexpensive followers and all proprietary models as expensive leaders would obscure the decisions buyers actually face.

KEY TERMS, AS USED IN THIS REPORT

Open-weight
"Describes access to a model's learned parameters."

Frontier
"Describes capability." An open-weight model "can be at or near the frontier on a particular task."

Open-source AI
The OSI definition "includes freedoms to use, study, modify and share a system."

Self-hosting
"The organization operates the model-serving stack on owned equipment or rented infrastructure."

Access to weights allows an organization to operate a checkpoint on infrastructure it controls, subject to the applicable terms and technical requirements. It can also create opportunities to adapt the model, preserve a working version and choose among serving providers. None of those rights should be inferred from a product's family name. The exact release and its license are the procurement unit, particularly when a publisher offers several variants through different channels.

Open-source AI makes a broader claim. The Open Source Initiative's definition includes freedoms to use, study, modify and share a system, along with the code, parameters and sufficiently detailed training-data information needed to exercise those freedoms. It does not simply require that every piece of training data be downloadable. Equally, distributing weights under a familiar license does not by itself establish that the complete system meets that definition. [3] Buyers can value open weights while being precise about the transparency they have actually received.

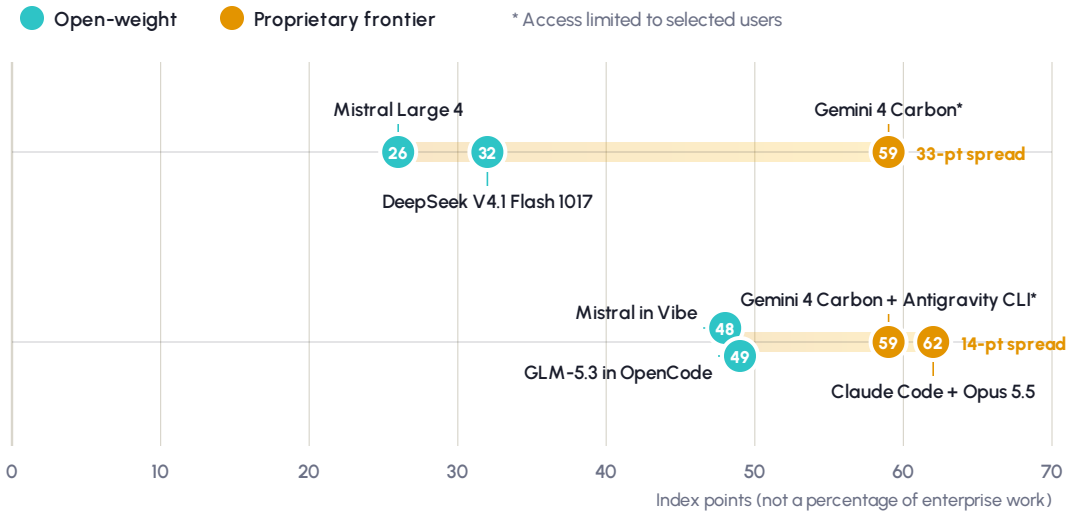
The difference is visible in the current offerings. Mistral's weights are downloadable, but its Mistral Ecosystem Open-Weight License includes a separate commercial-license condition for specified model-as-a-service businesses, with exceptions including internal use. GLM-5.3 uses a custom license with a security-review condition for qualifying model-as-a-service businesses above a stated revenue threshold. Neither should be casually described as unrestricted MIT. [4, 5, 6, 7]

Availability needs the same discipline as licensing. An announcement that weights will arrive next week is not a deployment option this week. A benchmarked service may

EXHIBIT 3

On broad indexes, the proprietary frontier still leads

Artificial Analysis Intelligence Index v5 and Coding Agent Index, late October 2026



Sources: Artificial Analysis Gemini 4 Carbon report (Oct. 28), Mistral Large 4 evaluation and DeepSeek V4.1 Flash 1017 evaluation. Composite index scores are not percentages of enterprise work. [12, 2, 13]

also include components that are absent from the downloadable package. Before treating a hosted evaluation as evidence for an internal installation, the buyer should establish exactly which checkpoint, precision, reasoning configuration and surrounding software produced the result.

02

Competitive at which tasks?

The proprietary frontier still deserves its name on demanding general evaluations. Artificial Analysis's October 28 report put Gemini 4 Carbon at 59 on Intelligence Index v5, while warning that access was limited to selected users. Its coding-agent comparison put Carbon with Antigravity CLI at 59, behind Claude Code with Opus 5.5 at 62. [12] Mistral Large 4 scored 26 on Intelligence Index v5, while DeepSeek V4.1 Flash 1017 scored 32. [2, 13] These are composite index scores, not percentages of all enterprise work a model can complete.

That snapshot establishes substantial differences on the evaluator's chosen workload. It does not mean that a document classifier receives twice the useful capability by moving from a score near 30 to one near 60. Nor does it establish that every task needs the top model. A composite index weights several kinds of work; an enterprise application may depend overwhelmingly on one narrow behavior and have no use for the rest. The score helps identify candidates, while the application determines whether the difference is commercially meaningful.

> *The score helps identify candidates, while the application determines whether the difference is commercially meaningful.*

Coding illustrates the problem. Completing a line in an editor, explaining an unfamiliar repository and autonomously resolving an issue are all coding-related activities, but they demand different systems. A useful comparison must account for repository access, test execution, context preparation, tool permissions and the amount of time or reasoning allowed. A model operating inside a well-designed agent can produce a better outcome than the same model exposed through a generic chat interface. The enterprise buys or builds that surrounding system too.

Artificial Analysis documents different setups for its evaluations. Its Intelligence Index uses mini-swe-agent for Terminal-Bench 4.0 and Stirrup for FrontierCode 2.0. Its Coding Agent Index evaluates complete agent variants and uses a different FrontierCode version alongside other tests. [14, 15] This prevents two common shortcuts: calling the broad index a bare-model test, and treating two scores with a similar benchmark name as directly comparable without examining the methodology.

SIDEBAR 01

Five questions to ask before calling a model open source

1

Can we obtain the actual weights?

Confirm that the files are available today and identify the exact checkpoint. API access, a research preview and a promise of a later release are different forms of availability. Retain the model card and the terms that accompany the downloaded version.

2

What does the license permit us to do?

Examine internal use, modification, redistribution and offering services to others. A license may distinguish an enterprise application from a model-as-a-service business. Familiar terminology on a launch page is not a substitute for reading the governing document.

3

What information accompanies the weights?

Determine whether training code, inference code and meaningful training-data information are available. Under the Open Source Initiative's definition, the package must support the freedoms to study and modify the system, not merely make inference possible. [3]

4

Can we reproduce the behavior we evaluated?

Preserve the configuration, tokenizer, precision, prompts and tools as well as the weights. If the published score came from an agent system, find out which components are included and which must be obtained or developed separately.

5

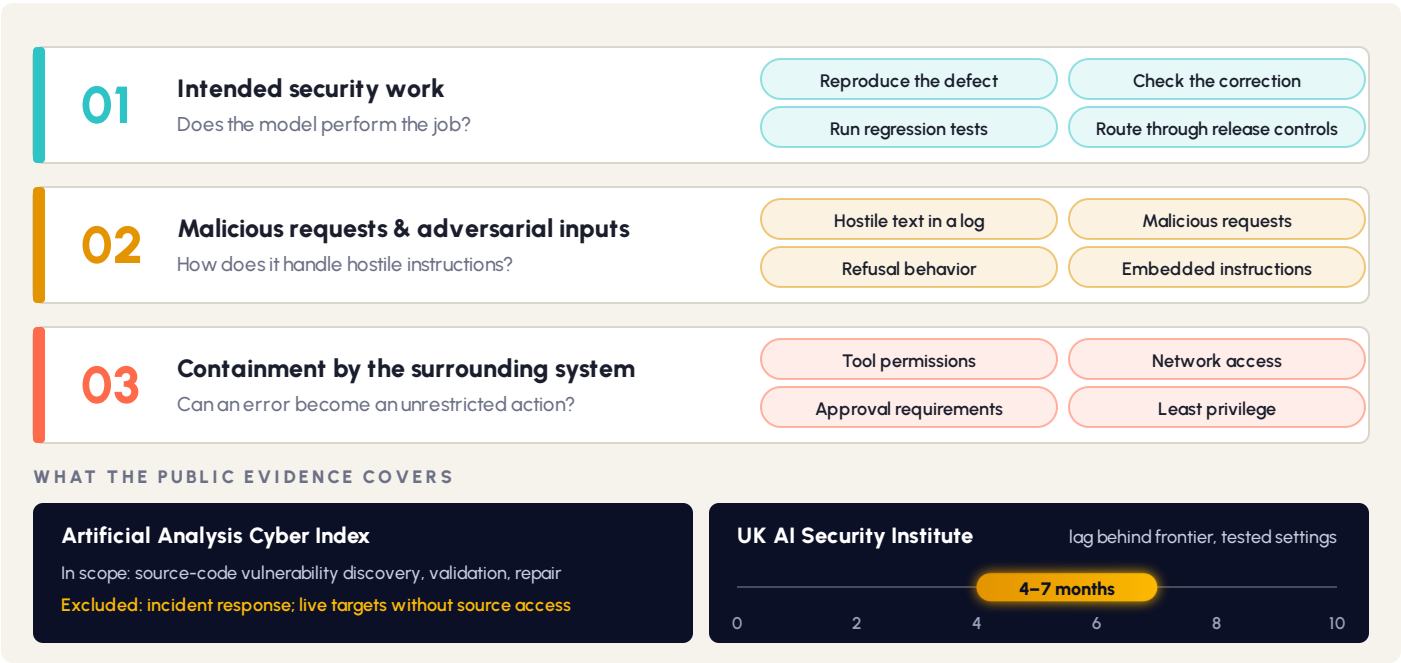
Can we keep operating and maintain the system?

Establish what happens if the original publisher or serving provider withdraws support. Possessing files is useful, but continuity also requires compatible software, infrastructure, operational knowledge and the applicable rights. Record these requirements in the exit plan before treating the model as an independence guarantee.

EXHIBIT 4

Cyber readiness needs three separate tests

The report's evaluation framework, set against what public cyber evidence covers



Framework: this report's recommendation. Evidence: Artificial Analysis Cyber Index methodology; UK AI Security Institute. [16, 17]

For document workflows, sufficient capability should be defined before model selection. A hypothetical invoice system might require correct extraction of named fields, a reliable signal when a field cannot be established, and evidence linking the answer to the source document. A fluent explanation that invents a missing value fails that task. Conversely, a small model that consistently extracts the required fields may be more useful than a large model that offers an elaborate interpretation nobody asked for.

Retrieval adds another layer. If an assistant receives an outdated policy or retrieves a document the user should not see, a stronger language model cannot by itself repair the information architecture. Evaluation should test the full path from permissions and document selection to the final answer. Buyers should also measure whether citations support the answer, whether the system recognizes conflicting information and whether it escalates a question that the available evidence cannot resolve.

The test should include ordinary work, awkward exceptions and cases where the right response is to stop. Use a held-out set that was not used to tune prompts, and keep the production workload represented as it changes. If a cheap model passes only on familiar templates, the apparent savings may disappear when a customer introduces a different form. If a premium model offers no measurable advantage on the accepted workload, its broader benchmark lead is not a sufficient reason to keep paying the premium.

There is also a purchasing constraint that rarely fits neatly on a leaderboard: access. A model available only to selected organizations cannot be assumed to be an immediate procurement choice. Benchmarked capability, commercial availability and the ability to operate the system yourself belong in separate columns. The same standard applies to a proprietary preview and an open-weight release whose files have not yet appeared.

03

Cybersecurity deserves three separate evaluations

Cybersecurity makes the stakes more obvious because a model can be useful to defenders, useful to attackers and vulnerable to manipulation at the same time. Those properties need separate tests. A system that finds a real vulnerability has demonstrated one capability; it has not thereby demonstrated that it can safely handle hostile instructions embedded in a log, limit its own

SIDEBAR 02

What the benchmarks answer

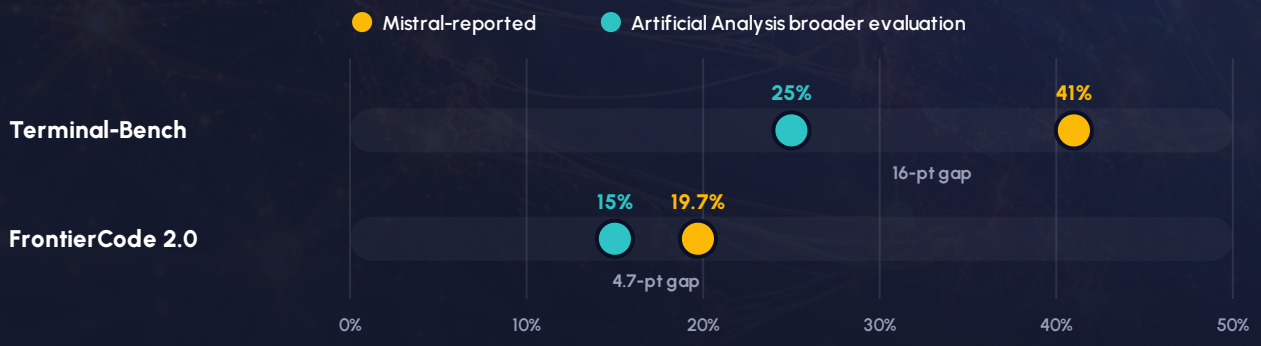
EVIDENCE	USEFUL QUESTION IT ADDRESSES	WHAT IT DOES NOT ESTABLISH
Broad intelligence index	How did the evaluated configuration perform across a defined collection of tasks?	The percentage of your enterprise work it can complete
Coding-agent benchmark	How did a model, agent and tool setup perform together?	The result from another harness, budget or repository environment
Source-code cyber evaluation	Can the system find, validate or repair vulnerabilities within the test?	SOC effectiveness, incident response or immunity to prompt injection
Serving benchmark	What throughput and latency did a particular system achieve under specified conditions?	Capacity for a different model, context distribution or service target
Internal acceptance evaluation	Does the proposed application meet a defined business requirement on the tested sample?	Permanent performance after the workload, data or model changes

Sources for the external evaluation categories: Artificial Analysis methodology and MLCommons. The application-specific acceptance guidance is this report's recommendation. [14, 15, 16, 33]

EXHIBIT 5

Same benchmark names, different reported results

Mistral Large 4 scores as published by Mistral and by Artificial Analysis; settings not reconciled



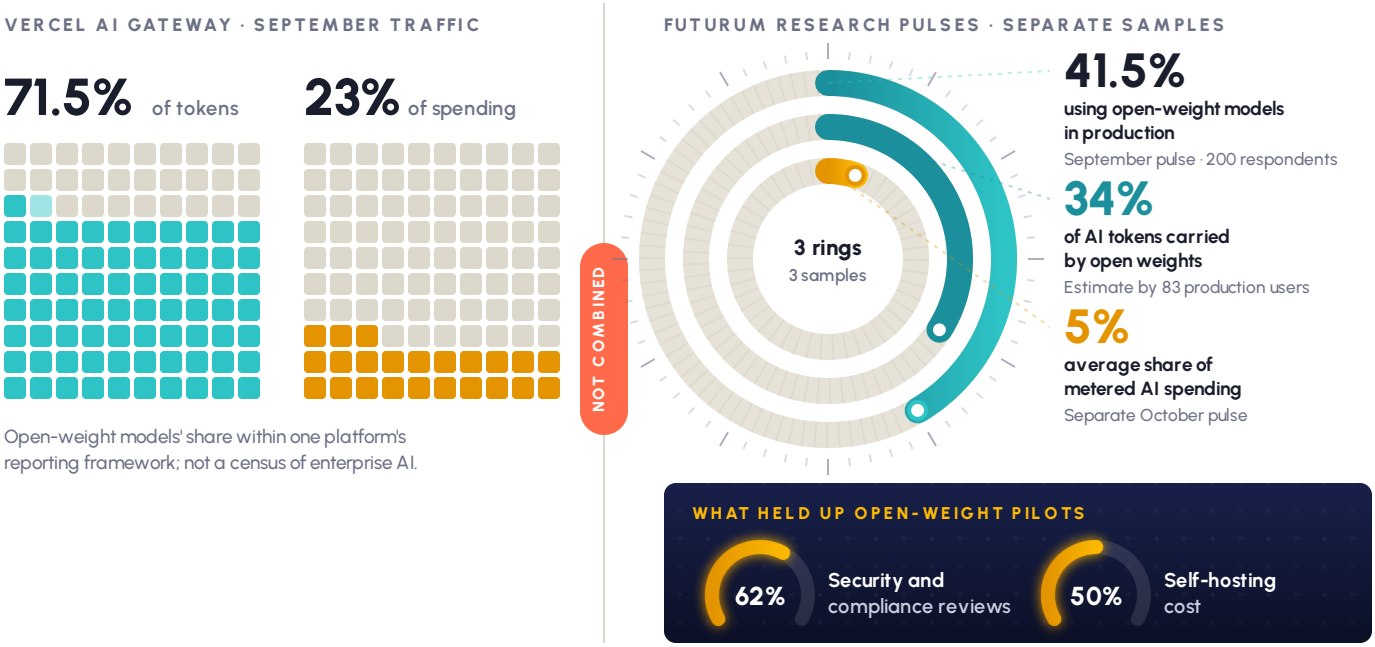
For the Mistral launch, the published results require special care. Mistral reports 41% on Terminal-Bench and 19.7% on FrontierCode 2.0, while Artificial Analysis reports 25% and 15% in its broader evaluation. [34, 2] Different configurations are a comparability concern; the research for this report did not fully reconcile the settings or establish the cause of each difference. Neither pair should be presented as the other's reproduction.

A procurement comparison should retain the date, benchmark version, checkpoint, reasoning setting, tool environment and execution budget. Where repeated runs or uncertainty estimates are available, examine them before treating a one-point lead as decisive. A result can be accurate within its test and still be unsuitable as evidence for a different purchasing decision.

EXHIBIT 6

Tokens and dollars tell different stories

Open-weight adoption measures from different samples, platforms and scopes, shown separately



Sources: Futurum open-weight and AI tools pulses, Sept.–Oct. 2026; Vercel AI Gateway Production Index, October 2026. Figures cannot be combined into a savings ratio. [18, 21]

permissions or produce a patch suitable for immediate deployment.

Artificial Analysis's Cyber Index addresses source-code vulnerability discovery, validation and repair. Its stated scope excludes incident response and live targets without source access. [16] That is valuable evidence for software security work, but it does not establish how an enterprise SOC will perform during an intrusion. Incident response requires reasoning across incomplete telemetry, understanding local business context and taking actions whose consequences extend beyond a benchmark environment.

The UK AI Security Institute provides another useful, bounded perspective. Its study of leading open-weight models found a roughly four-to-seven-month lag against the proprietary frontier in the cyber settings it tested. The evaluations included narrow tasks and simulated longer attack sequences; the latter lacked active defenders and some real-world protections. [17] The result is historical evidence about specified models and environments, not a permanent timetable for open-weight progress or an October rating of Mistral's new release.

The enterprise evaluation should therefore have three tracks. Test whether the model performs the intended security work. Test how it handles malicious requests and adversarial inputs. Then test whether the surrounding system prevents an error or compromise from becoming an unrestricted action. Refusal behavior can contribute to the second track, but it should not substitute for the third. Tool permissions, network access and approval requirements are operational controls that the organization must design deliberately.

For a patching workflow, completion should mean more than generating a plausible diff. The process should reproduce the defect where appropriate, check the proposed correction, run regression tests and route the change through the organization's release controls. For an alert-triage workflow, the evaluation should measure missed threats and unnecessary escalations against an explicit baseline. Neither purchase price nor model nationality answers those questions.

> *Security responsibility follows the application and the actions it can take.*

I would apply that standard to every provider. A locally operated model can reduce exposure to an external inference service, but it can still be given excessive privileges. A proprietary service can provide managed controls without understanding every customer's internal authority structure. Security responsibility follows the application and the

actions it can take. Selecting the model is one decision within that responsibility, not a way to discharge it.

04

Adoption is becoming a portfolio decision

Futurum's September open-weight pulse found 41.5% of 200 respondents using such models in production. Its 83 production users estimated that open weights carried 34% of their AI tokens. A separate October pulse put their average share of metered spending at 5%. Those figures describe different samples and scopes; they cannot establish a matched token-to-dollar savings ratio. [18]

The direction is useful even without forcing the numbers into an economic claim. Enterprises are finding production uses for open models while retaining other AI purchases. An organization can run a specialized internal model, subscribe to a general assistant and send difficult tasks to a premium API without contradicting its strategy. Each purchase serves a different combination of workload, access and operational requirements.

Other research supports coexistence. Amplify Partners and Notion's shared engineering survey reports that more than 90% of open-weight users also use closed models. Their public presentations disagree on some other headline percentages, so those disputed figures are excluded here. [19, 20] This is one study presented in two places, not two independent confirmations. Still, the overlap argues against treating open-weight adoption as synonymous with abandoning proprietary AI.

Vercel's October AI Gateway Production Index provides a different kind of evidence. For September traffic through its platform, open-weight models represented 71.5% of tokens and 23% of spending. [21] Unlike the two Futurum figures, these describe activity within one platform's reporting framework. They show that token volume and spending can diverge sharply, but the platform's customers and workload mix do not constitute a representative census of enterprise AI.

Survey responses, platform telemetry and customer accounts answer different questions. Surveys can expose buying priorities and deployment intentions. Telemetry records activity within a particular service. Customer accounts can explain why a specific workload moved. Combining them is useful when their boundaries remain visible; adding their percentages together or presenting them as interchangeable market shares creates certainty the evidence does not support.

Futurum also reports that security and compliance reviews held up 62% of open-weight pilots, while self-hosting cost held up 50%. [18] These are reasons to examine implementation requirements, not proof that open weights are inherently less secure or less economical. A pilot can stall because an organization has not established a review process, has selected an unnecessarily large model or lacks the staff to operate its preferred architecture. The survey identifies friction without isolating its cause.

For buyers, a portfolio approach creates its own responsibilities. Somebody must maintain the evaluations, decide which requests go where and know what happens when a provider or model becomes unavailable. More choices can improve negotiating leverage, but an ungoverned collection of endpoints adds support work and inconsistent behavior. The organization needs enough flexibility to replace a model without making every application responsible for solving procurement, security and monitoring independently.



05

Sovereignty requires a dependency inventory

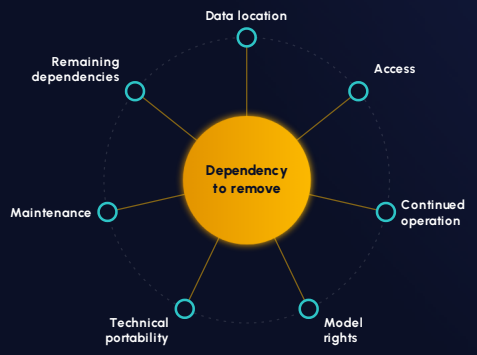
Possession of a model can solve a real continuity problem. If an organization has the weights, the required software, the infrastructure and the necessary rights, the publisher's decision to withdraw a hosted endpoint does not automatically stop that installation. The organization can preserve a version that has passed its own evaluation rather than accepting a provider's upgrade schedule. That can be worth paying for even when the self-hosted option costs more.

Residency is a related but separate concern. Keeping inference in a particular location does not establish where every supporting service processes data. Retrieval, observability, authentication, backups and external tools all belong in the review. A model can run locally while its agent sends sensitive information to another service. Conversely, a managed offering may satisfy a specific residency

SIDEBAR 03

A practical sovereignty checklist

Write down the dependency the proposed architecture is intended to remove. Then document the following responsibilities and failure cases:



AREA	EVIDENCE TO REQUEST
Data location	Inference, retrieval, logs, backups and tool-service processing locations
Access	Which organizations and personnel can administer systems or access data
Continued operation	Behavior if the model publisher, cloud provider or external tool becomes unavailable
Model rights	Terms covering the chosen use, modification, retention and redistribution
Technical portability	A tested alternative serving environment and the effort required to move
Maintenance	Ownership of security updates, checkpoint evaluation and rollback
Remaining dependencies	Accelerators, software, identity services and other required suppliers

Use this inventory to decide what control is worth buying. An on-premises deployment might satisfy a requirement for local possession while retaining substantial supplier dependencies. A managed regional service might satisfy a residency requirement while leaving continuity dependent on the provider. Those can both be reasonable choices if the enterprise states which requirement it is meeting and accepts the remaining exposure explicitly.

SIDEBAR 3

requirement without giving the customer possession of the weights.

Technological independence is broader still. An organization may control inference while depending on imported accelerators, a particular software stack and a publisher for improvements to the next model generation. Fine-tuning a checkpoint does not necessarily provide the knowledge or resources to reproduce its original training. These dependencies do not cancel the value of local control, but they should be recorded rather than hidden behind a sovereignty label.

The same examination should apply to European, Chinese and U.S. suppliers. Country of origin can influence contractual and procurement requirements; it does not replace testing the system or mapping its data flows. The useful question is which dependency the enterprise needs to remove, which ones it can tolerate and what it will pay to manage the remainder. A sovereignty requirement that cannot name the dependency it addresses is difficult to translate into a defensible architecture.

Open weights expand the available answers. They do not require the organization to choose the most labor-intensive deployment. A buyer may use managed inference now, preserve the option to move later and periodically test whether that option is technically and contractually real. The ability to move has value only if the application, data and operating knowledge can move with it.



PART II

What It Costs to Put the *Model* to Work

06 Four purchases that should not be confused

07 What it takes to host a model

08 Three workloads, three economic decisions

09 The variables that determine the result

10 What customer experience tells us

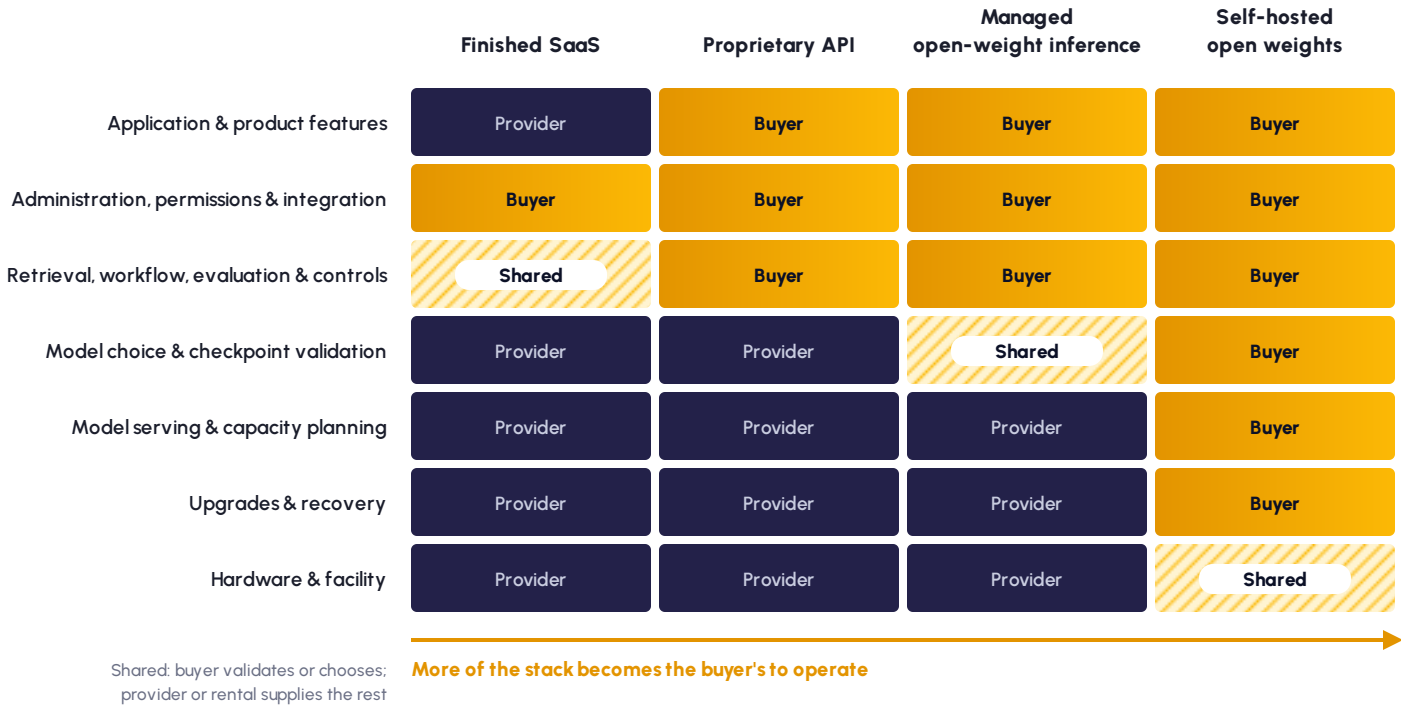
EVIDENCE NOTE: PRICES ARE OBSERVED; WORKLOADS ARE ILLUSTRATIVE

The prices below are public list prices checked on October 29, 2026, in U.S. dollars before tax. They are not negotiated quotes, guarantees of available capacity or measured total-cost results. The three workloads are hypothetical. Their token volumes, staffing sensitivities and review assumptions are disclosed so readers can replace them with their own data. Customer-reported results appear separately. No scenario assumes equal model quality or assigns an unmeasured accuracy advantage to either open or proprietary models.

EXHIBIT 7

Four purchases, four different operating burdens

Who principally supplies or operates each layer, as described in this report



Shared: buyer validates or chooses; provider or rental supplies the rest

PURCHASE	WHAT IT PRINCIPALLY SUPPLIES	RESPONSIBILITIES THE BUYER MUST ACCOUNT FOR
Finished SaaS	An application with bundled model access and product features	Administration, adoption, permissions, integration and workload suitability
Proprietary API	Hosted access to a provider's model	Application, retrieval, workflow, evaluation and operational controls
Managed open-weight inference	Provider-operated access to selected open models	Application responsibilities, checkpoint validation and provider terms
Self-hosted open weights	Customer-operated model serving	Application responsibilities plus serving, capacity, upgrades and recovery

Source: report analysis. Seat prices: ChatGPT Business Standard and Claude Team Standard list $20 per seat per month on annual billing or $25 monthly. [22, 23]

06

Four purchases that should not be confused

A finished AI application buys a substantial amount of work already done. Depending on the product and plan, the customer receives an interface, account administration, file handling, connectors and access to models. The enterprise still has to manage adoption, permissions and the suitability of the application for its work. But it is not starting with an inference endpoint and asking an engineering team to recreate the rest.

An API purchase is different. It supplies a model capability that developers can incorporate into a workflow. The organization must still provide the application logic, data access, error handling and operating controls, whether it develops those components or buys them elsewhere. Managed open-weight inference occupies a similar position: the provider operates the serving infrastructure, while the customer retains responsibility for the application. Open weights do not automatically mean self-hosting.

Self-hosting adds another layer of responsibility. Here, the organization operates the model-serving stack on owned equipment or rented infrastructure. Renting a GPU removes hardware ownership and facility management from the customer's immediate workload, but it does not necessarily remove software maintenance, capacity

planning or availability responsibilities. The dividing line is who operates the inference service, not whose name appears on the building.

> *The dividing line is who operates the inference service, not whose name appears on the building.*

This is why a seat price cannot be compared directly with the token charge for the same user's prompts. The difference may be paying for useful software rather than overpriced inference. The reverse is also true: a subscription designed for individual interaction may not deliver the unattended pipeline a department needs. Buyers should compare complete solutions to the same problem and then examine which components they can obtain more economically elsewhere.

The category names also conceal pricing variation. ChatGPT Business Standard and Claude Team Standard list $20 per seat per month on annual billing or $25 monthly. OpenAI states that API usage is separate from ChatGPT Business. Claude's page advertises a limited pooled API-credit benefit for Team, which should be checked against the offer's terms rather than treated as unlimited API access. Enterprise arrangements and usage allowances require their own comparison. [22, 23]

07

What it takes to host a model

The first hosting decision should be an acceptance specification. Define the work, the permitted data, the response-time target, the expected peak concurrency and the conditions under which the system must decline or escalate. Then select candidate checkpoints and establish their serving requirements. Starting with a GPU purchase and looking for enough work to justify it reverses the process and makes idle capacity somebody else's problem after the budget has been approved.

A practical deployment needs a compatible inference engine, supported model precision and an installation that can be reproduced. Teams should retain the checkpoint identifier, serving configuration, tokenizer and other relevant dependencies alongside evaluation results. If an upgrade changes behavior, the operator needs to know what changed and how to return to the previous working version. A model endpoint that works once on a developer's machine is not yet an operated service.

Memory is one constraint; usable throughput is another. Model weights share accelerator memory with runtime requirements and the state needed to process requests. Longer contexts and more simultaneous requests can change the capacity calculation substantially. Quantization can reduce weight memory, but it must be checked for serving compatibility, performance and task quality. Parameter count alone cannot tell a buyer how many concurrent users a configuration will support. [24]

Mistral's largest release makes the scale concrete. The vLLM recipe lists validated deployments using eight B200 GPUs for NVFP4 and 16 H200 GPUs for FP8. [25] These are published working configurations, not universal minimums. A smaller model presents a different infrastructure problem, which is why evaluating a nine-billion-parameter candidate and evaluating Le Chonk should not begin with the same purchasing assumption.

> *A model endpoint that works once on a developer's machine is not yet an operated service.*

After serving comes integration. The application needs an interface or API, permission-aware retrieval where appropriate, credentials, logging and error handling. An agent also needs a controlled environment for executing tools. CPU time, storage, external services and databases belong in the capacity plan. The ability to generate a tool call does not supply a safe tool runtime, and a GPU bill does not cover every action the resulting application takes.

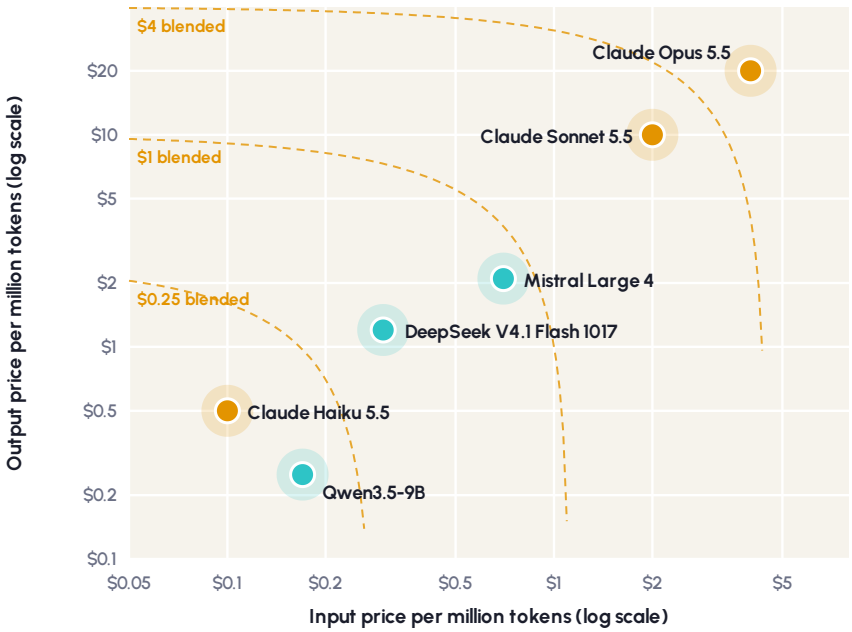
Availability requires testing rather than a standard hardware multiplier. Decide which failures the service must survive, how long recovery may take and whether a temporary fallback is acceptable. A business-hours internal assistant and a customer-facing service have different requirements. An availability percentage by itself does not establish a particular node count; workload peaks, maintenance, failure domains and recovery procedures determine what the operator must provision.

Finally, assign ownership. Somebody must respond to degraded latency, investigate a quality regression, update the serving stack and determine whether a new checkpoint should replace the old one. An organization with an established inference platform may absorb those tasks efficiently.

EXHIBIT 8

A price map that includes inexpensive proprietary inference

Published rates per million tokens, input vs. output; dashed curves show blended price at a 90/10 input/output mix



PUBLISHED RATES	IN	OUT
Claude Haiku 5.5 Anthropic	$0.10	$0.50
Claude Sonnet 5.5 Anthropic	$2.00	$10.00
Claude Opus 5.5 Anthropic	$4.00	$20.00
Qwen3.5-9B Together	$0.17	$0.25
DeepSeek V4.1 Flash 1017 Together	$0.30	$1.20
Mistral Large 4 First-party, via Artificial Analysis	$0.70	$2.10

Sources: published pricing and evaluator pricing records. Haiku rates assume prompts no longer than 100,000 tokens. Calculations exclude caching and batch discounts, promotional credits, tools, regional premiums and taxes. All reasoning, retries and repeated context must fit within the stated billable volume; otherwise the volume must increase. [23, 26, 27]

One without that capability should account for building or buying it. Existing staff can reduce incremental hiring, but their time still has an opportunity cost.

08

Three workloads, three economic decisions

The following price inputs deliberately include inexpensive proprietary inference as well as open-weight endpoints. Comparing only a premium proprietary model with a cheaper open model would make the licensing category appear to explain a price difference that may instead reflect model size, capability or service positioning. These are selected examples, not a recommendation that every listed model can successfully perform every scenario.

Scenario A: A small-team assistant

Consider 25 employees, with 15 active on a typical workday. At 20 tasks per active user across 22 days, the application handles 6,600 tasks a month. Assume 6,000 input and 800 output tokens per task, yielding 39.6 million input tokens and 5.28 million output tokens. This is a hypothetical drafting and summarization workload, with no assertion that the same prompt or token budget produces equal results across models.

At the Standard seat prices above, 25 SaaS seats represent $500 per month on an annual-equivalent basis or $625 on monthly billing, before additional consumption and administration. That does not establish entitlement to this exact workload. The buyer must check features and usage allowances. It does establish that the difference between a raw model bill and a product subscription is small enough to be overtaken by modest amounts of engineering or support work.

For this team, the initial question should be whether an existing application meets the need. Building an internal assistant to save a few hundred dollars in subscriptions requires a particularly clear explanation of the development and maintenance burden. Building because the team needs a unique workflow, offline operation or a specific data boundary is a different justification. Those benefits can be worth buying even when they do not reduce the monthly bill.

A rented GPU is another distinct commitment. Lambda lists an A10 configuration at $1.29 per GPU-hour, or about $942 across 730 hours, and an H100 PCIe configuration at $3.29, or about $2,402. [28] Neither figure establishes that the

SIDEBAR 04

Fitting a model in memory is only the beginning

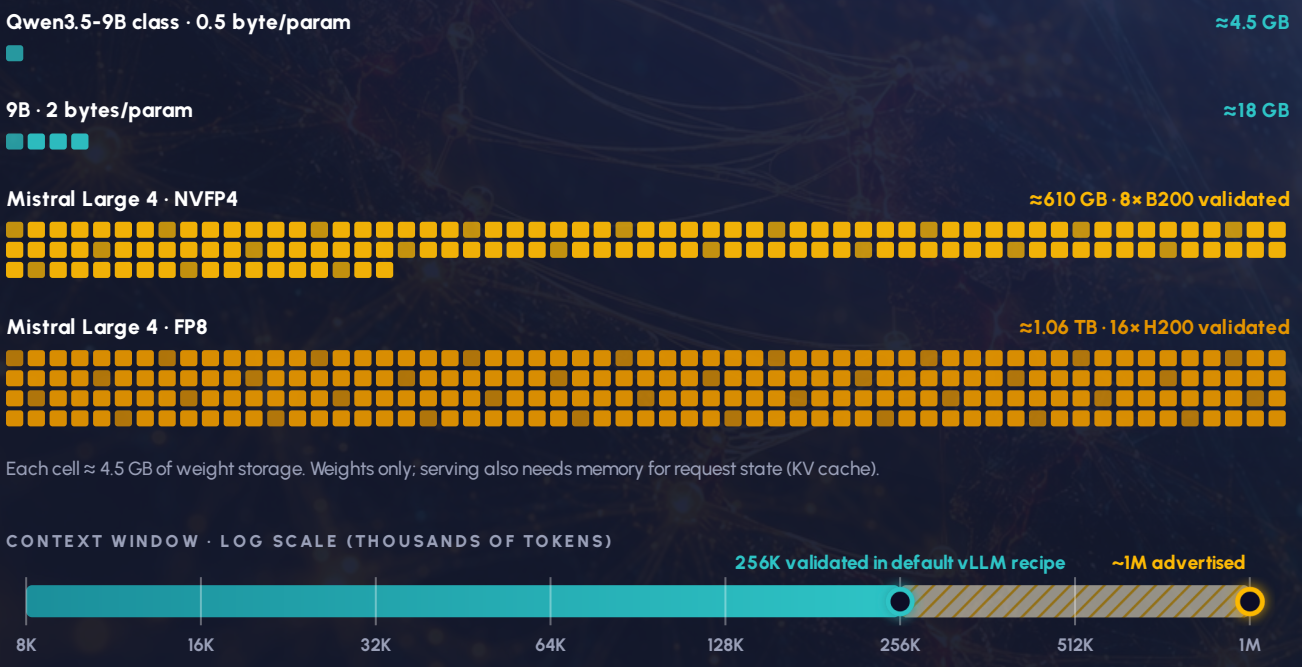
Parameter count gives a first approximation of weight storage. Nine billion parameters at two bytes each imply approximately 18 GB before runtime overhead; at half a byte each, the arithmetic implies 4.5 GB. Actual quantized checkpoints include metadata and may retain some components at higher precision. These calculations estimate weight storage, not the complete memory required to serve requests.

Serving also needs memory for request state, including the attention cache often called the KV cache. Its size depends on the model architecture, context lengths, precision and simultaneous requests. A generic calculation for one architecture should not be transferred to another without checking those details. The configuration that loads successfully may still fail under the concurrency expected in production. [24]

EXHIBIT 9

From a laptop-scale model to a multi-GPU node

Approximate weight storage by model and precision; validated configurations from the vLLM recipe



Sources: vLLM Mistral Large 4 serving recipe; report arithmetic for 9B examples. [24, 25]

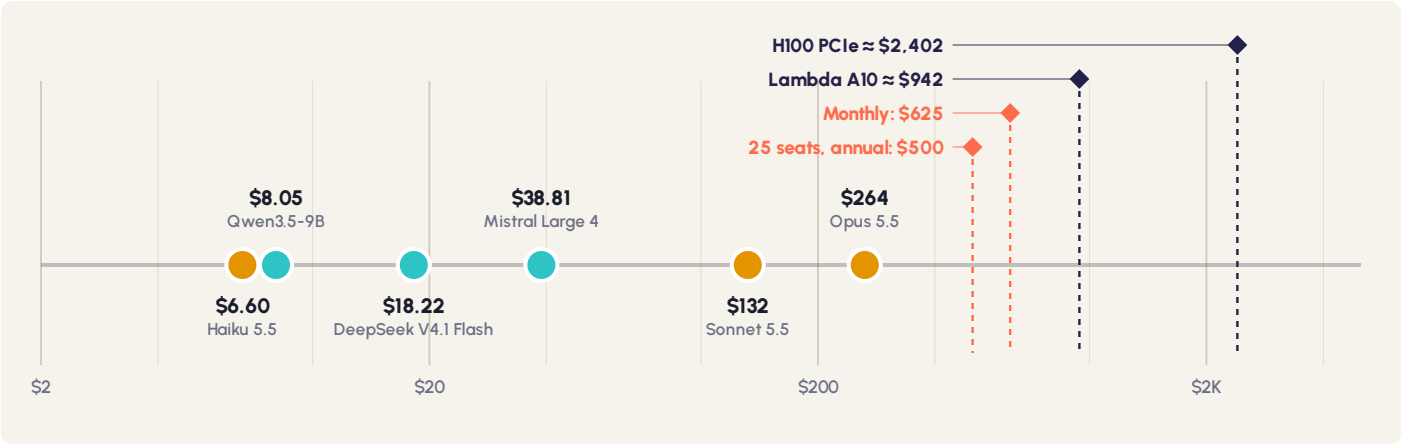
Mistral Large 4's published vLLM recipe illustrates the gap between a rough parameter calculation and a supported setup. It lists approximately 610 GB for the NVFP4 checkpoint and 1.06 TB for FP8, with validated examples on eight B200s and 16 H200s respectively. The default recipe validates context to 256,000 tokens even though the model advertises roughly one million. The H200 example does not supply performance figures. [25]

Input processing and output generation also place different demands on the system. Measure time to first token, output speed, total completion time and completed tasks at the expected concurrency. A batch system can trade waiting time for efficiency more readily than an interactive assistant. The deployment decision should be based on the resulting service, not simply the fact that the weights fit.

EXHIBIT 10

At small-team scale, every model bill sits below the seat price

Scenario A: calculated monthly model charge, 39.6M input + 5.28M output tokens (log scale)



Calculations from published rates; 25 Standard seats at $20 annual-equivalent or $25 monthly; Lambda A10 at $1.29 and H100 PCIe at $3.29 per GPU-hour over 730 hours. Hypothetical workload. [23, 26, 27, 28]

configuration can serve the chosen model at the required latency. Turning equipment off outside working hours or using an existing workstation may improve the calculation, but the application still needs an operator. At this scale, the strongest self-hosting case is likely to involve a specific control requirement or existing capacity, rather than token savings alone.

Scenario B: A departmental document workflow

Now consider a department processing 4,000 documents per business day, or 88,000 a month. Assume extraction, validation and any included retry allowance consume 30,000 billable tokens per document, with 90% input and 10% output. Monthly volume is therefore 2.376 billion input tokens and 264 million output tokens. The workload is illustrative; the appropriate model must be selected through a representative document evaluation.

This is an application project, not simply a seat purchase. The system must ingest documents, perform any required OCR, preserve evidence, manage exceptions and write accepted results into a business system. The cost of those functions exists with either a proprietary or an open model. A comparison should isolate the components that actually change when the model or inference provider changes, rather than charging the entire application to one option and treating it as free in another.

Review economics can change the preferred model quickly. Suppose an additional one percentage point of documents requires a four-minute review at an assumed labor cost of $50 an hour. That adds 880 reviews and approximately $2,933 per month. This is a sensitivity calculation, not a measured difference between models. It shows why a lower token charge can coexist with a higher total cost if the selected model creates enough extra work.

The same logic applies in the other direction. If a specialized open model reduces exceptions, its value can exceed the inference savings. Evaluation should measure incorrect accepted fields as well as flagged documents: a system that appears efficient because it confidently passes bad records has not solved the problem. The department needs an acceptance policy, a representative test set and an ongoing sample of production output before it can attach a credible value to the price table.

Self-hosting becomes worth investigating when throughput, existing capacity or control requirements justify it. It is not established by the document count alone. Long inputs, batch deadlines and interactive requests stress a serving system differently. Before committing to a server, replay representative work and measure the capacity available while meeting the required latency. Sharing infrastructure with other workloads may improve utilization, but only if their peaks and service commitments are compatible.

Scenario C: A large deployment with steady demand

For a larger platform, assume two million tasks per month and a rounded total of 70 billion billable tokens, again 90% input and 10% output. This could represent multiple calls within each task, but the routing calculation below is based on token volume, not the number of calls. Confusing those measures would make a short classification request and a long reasoning session count as equivalent demand.

An illustrative monthly budget that exposes the assumptions

The document scenario processes 88,000 documents monthly. To demonstrate how to build a complete comparison, assume a hypothetical custom application incurs $3,000 in shared application services, half an FTE at the report's illustrative $20,000 monthly rate, and $120,000 in setup costs amortized over 24 months. Those assumptions contribute $18,000 per month before inference or human review. They are not supplier quotes or a recommended staffing plan.

EXHIBIT 11

Review labor, not inference, dominates this budget

Hypothetical monthly cost build for 88,000 documents



COMPONENT	HYPOTHETICAL MONTHLY AMOUNT	EVIDENCE TYPE
Application services	$3,000	Assumption
Half an FTE	$10,000	Assumption
Setup amortization	$5,000	Calculation from assumed $120,000 over 24 months
Sonnet inference	$7,392	Published rates × scenario token volume
Human review	$29,333	Assumed 10% of documents × four minutes × $50/hour
Total	$54,725	Illustrative calculation, not observed TCO

If every document ultimately meets the acceptance requirement, that example costs approximately $0.62 per accepted document. If some remain unresolved, use the smaller accepted-result count and report the unresolved workload. The example excludes downstream loss from undetected errors, which requires a separate risk assessment.

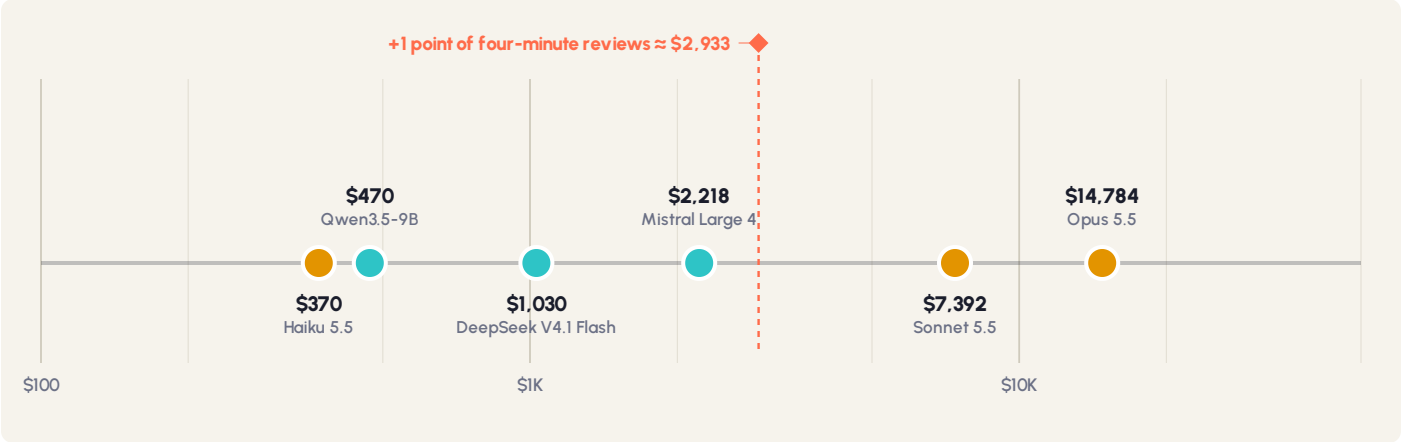
Changing only the inference line to the DeepSeek endpoint reduces the calculated total to approximately $48,363, or $0.55 per document under the same acceptance assumption. This is not a claim of equal performance. About 2.17 additional percentage points of four-minute human reviews would consume the $6,362 inference difference. A self-hosted comparison would replace the inference line with measured serving cost and add any genuinely incremental platform effort, rather than assuming the existing half-FTE covers it.

The purpose is to expose the variables, not declare a winner. A reader can replace every assumption with an observed cost, retain the accounting structure and see whether the conclusion survives.

EXHIBIT 12

One extra point of reviews costs more than most model bills

Scenario B: calculated monthly model charge, 2.376B input + 264M output tokens (log scale)



Calculations from published rates. Review sensitivity: 880 extra four-minute reviews at an assumed $50 per hour. Hypothetical workload. [23, 26, 27]

At this volume, model selection can move substantial money, but routing may matter more than committing the entire platform to one provider. Under the same input/output mix, sending 80% of token volume to Haiku and 20% to Sonnet produces a calculated model charge of $47,040. That is an illustration of price arithmetic, not evidence that the cheaper tier can successfully handle that share of the work. The routing policy itself needs evaluation, including the cost of recognizing when to escalate.

The quality sensitivity is equally substantial. Across two million tasks, one additional percentage point of failures represents 20,000 tasks. At an assumed $2 fallback cost each, that is $40,000 a month. The figure is not a universal price for human intervention, which could be much higher for complex work. It demonstrates why routing should optimize accepted outcomes and escalation costs, rather than merely increasing the share of calls sent to the least expensive endpoint.

Dedicated infrastructure deserves serious consideration here, but it needs a measured capacity model. Lambda's published eight-H100 SXM configuration calculates to about $23,302 for 730 hours, while its eight-B200 configuration calculates to about $39,070. [28] These are different instances with different capabilities. Neither price can be declared sufficient for this workload without testing the selected model, context lengths, concurrency and service targets. Recovery capacity, staff and application services remain additional considerations.

The large deployment's advantage is that it can spread platform investment across considerable work. Its risk is that a small error in assumptions becomes expensive at the same scale. A business case should test sustained demand, peak demand and a lower-growth case, then ask how quickly the architecture can adapt if API prices fall or a smaller model becomes adequate. Long-lived infrastructure commitments should not depend on today's premium model price remaining unchanged.

09

The variables that determine the result

Utilization deserves a precise definition. A GPU can be busy while the application misses its response-time target, and aggregate token throughput can look impressive while individual users wait too long. Capacity for a financial comparison should mean measured work completed at the required quality and latency. Input-heavy batch processing and output-heavy interactive generation cannot be treated as identical simply because both are counted in tokens.

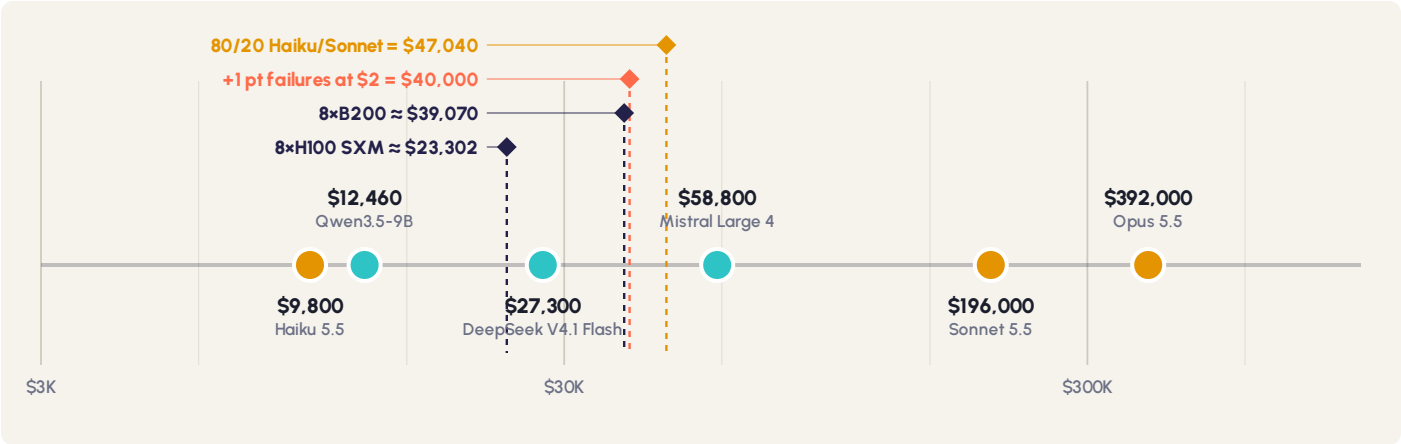
There is consequently no universal 50% utilization threshold at which self-hosting becomes economical. The crossover depends on the actual infrastructure bill, serving performance, alternative API price and operating burden. It can occur below that level, above it or not at all for the proposed configuration. The worksheet accompanying this report provides a simplified formula, but its most important input is measured capacity under the intended workload, not a supplier's maximum-throughput headline.

Staffing should be equally explicit. At an illustrative loaded cost of $20,000 per full-time-equivalent month, one-tenth of an FTE adds $2,000, half adds $10,000 and one

EXHIBIT 13

At platform scale, routing and failure rates are five-figure decisions

Scenario C: calculated monthly model charge, 70B billable tokens at 90/10 input/output (log scale)



Calculations from published rates; Lambda eight-GPU configurations over 730 hours; failure sensitivity at an assumed $2 per task. Price arithmetic, not evidence of equal quality. [23, 26, 27, 28]

adds $20,000. This is an assumption for sensitivity analysis, not compensation research. Charge the effort the architecture actually requires. Do not assume an entire new team is necessary for every small deployment, and do not assume that using an existing team makes the work free.

Caching, batching and shorter prompts can improve either open or proprietary economics. Their savings depend on the service's billing rules and the application's behavior. A repeated prompt that reliably hits a cache is different from a workload where every document is new. Similarly, an asynchronous batch discount is valuable only if the business can tolerate the associated delivery time. An honest comparison applies available optimizations to each candidate rather than giving one architecture an optimized implementation and the other a deliberately expensive baseline.

The accounting should distinguish cash from cost. An owned server requires an upfront payment; amortization spreads that expenditure across an assumed useful life. Those are different views, and adding both in the same period double-counts the hardware. A rented instance already incorporates the provider's facility expenses, so the customer should not add a hypothetical electricity and colocation bill for that same instance. First-year cash should reflect actual timing rather than automatically combining all setup costs with twelve months of full operation.

The result worth comparing is the cost of accepted work, including the attempts that failed. Count review, retries, escalation and remediation consistently. Then examine whether faster completion, better quality or stronger control creates additional business value. Cost per successful outcome is a stronger operating measure than cost per token, although it still needs an assessment of the consequences of the mistakes that remain undetected.

> *Cost per successful outcome is a stronger operating measure than cost per token, although it still needs an assessment of the consequences of the mistakes that remain undetected.*

10

What customer experience tells us

Gumloop offers a concrete example of changing models within an application. In its own July account, the company described using 72% fewer Gumloop credits after switching from Opus 4.8 to GLM-5.2. [29] The claim is useful because it identifies a workflow-level consumption difference, but credits are not a complete operating-cost measure. The account does not provide a controlled evaluation of review effort or a full accounting of the transition. Its strongest contribution is showing a model substitution that the user considered practical.

EliseAI illustrates specialization. In Baseten's customer account, EliseAI used a fine-tuned four-billion-parameter model and reported roughly 60% lower inference costs. The

SIDEBAR 05

A worksheet for comparing your own workload

Start with one task category and complete the following inputs for each candidate deployment. Use the same acceptance criteria across candidates, while measuring the attempts and tokens each actually consumes.

INPUT	RECORD
Workload	Monthly tasks, peak concurrency, context distribution and input/output mix
Acceptance	Required result, permitted errors, escalation rules and review sample
Model usage	Input, output and reasoning tokens; retries; cache writes and hits
Infrastructure	Required configuration, measured capacity, idle periods and recovery capacity
Operating effort	Incremental engineering, administration, evaluation, security and support
Human work	Review time, correction, fallback handling and remediation
Setup	Integration, labeling, tuning, testing and migration costs
Contract	Discounts, minimum commitments, usage limits and exit costs

FOR A MONTHLY OPERATING COMPARISON:

Total cost = model charges or allocated serving infrastructure + application services + incremental operating effort + security/support + review/remediation + amortized setup.

Cost per accepted result = total cost of all attempts, including failures, divided by accepted final results.

Track rejected or unresolved work separately so a system does not look efficient merely by declining difficult tasks. Keep cash-flow analysis alongside the operating comparison: equipment purchases and annual prepayments occur when paid, while amortization is a different accounting view. Do not count both as separate hardware expenses in the same cost calculation.

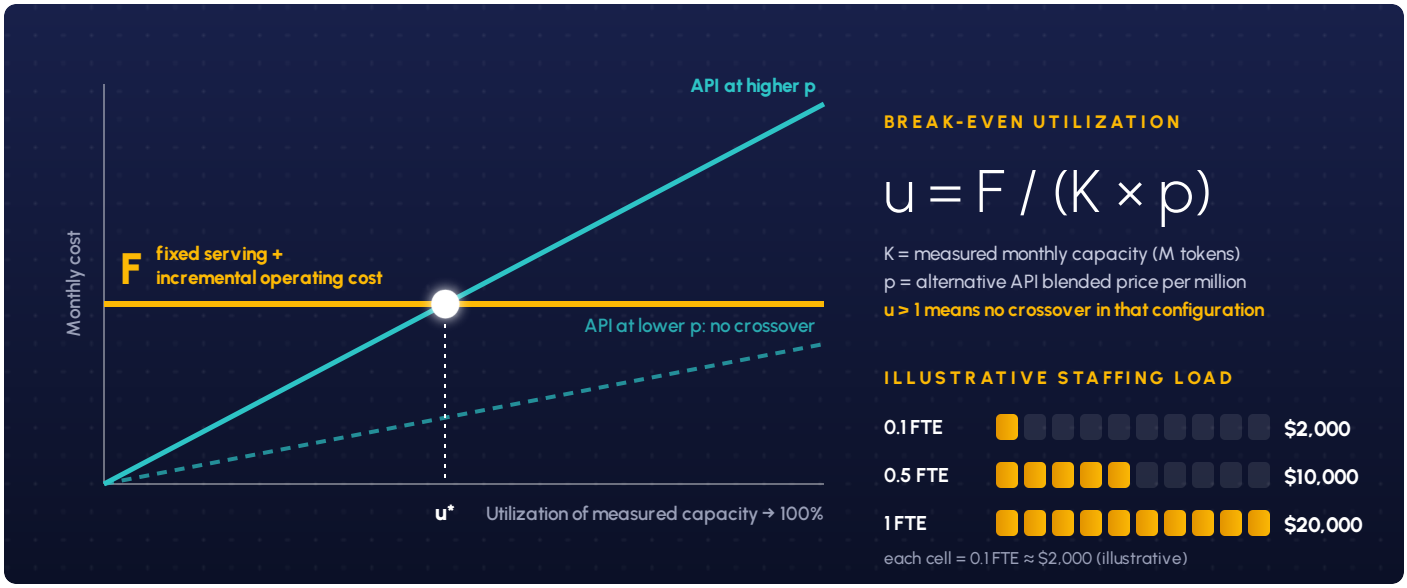
A simplified self-hosting crossover can be calculated if the workload mix and quality are matched. Let F be monthly fixed serving and incremental operating cost, K be measured monthly capacity at full useful utilization in millions of workload-equivalent tokens, and p be the alternative API's blended price per million. Ignoring variable hosting costs, the break-even utilization is F divided by K times p: $u = F / (K \times p)$. A result above one means no crossover within that configuration's capacity under these assumptions.

The formula is a screening tool, not a universal threshold. Real systems have variable costs, capacity steps, queueing limits and different token consumption by model. Common application costs can cancel in an incremental comparison; costs that change must remain. If sufficient throughput has not been measured at the required latency and quality, the business case is not ready to claim a hosting advantage.

EXHIBIT 14

There is no universal utilization threshold

Conceptual crossover for self-hosting against an API alternative, with illustrative staffing sensitivity



Formula and staffing assumption from this report's worksheet (Sidebar 5). Conceptual diagram, not to scale.



case describes 99% accuracy for one feature and p90 latency of 250 milliseconds compared with 2.2 seconds. [30] These are provider-published results for particular work, not evidence that a four-billion-parameter model broadly replaces the frontier. They do support testing whether a narrowly trained model can deliver the behavior an application needs more efficiently. Training, labeling and full ownership costs remain outside the disclosed comparison.

Sourcegraph supplies an instructive counterweight to the assumption that model control requires infrastructure ownership. Fireworks' January 2025 case says self-hosting was considered less viable because of operational complexity and scalability. Sourcegraph's own documentation confirms Fireworks use in Cody. [31, 32] That corroborates the deployment choice, although it does not independently validate every performance claim in the provider's case study. It is a documented decision to use managed inference, not a failed self-hosted rollout or a complete cost audit.

Together, these examples describe different optimization opportunities: change the model, specialize the model or change who operates it. None requires an ideological commitment to an entirely open or entirely proprietary stack. They also reveal what published customer stories usually leave unanswered in the evidence reviewed here: the total engineering effort, the cost of evaluation, the treatment of failures and the economics of the alternative that was not deployed.

We did not locate an independently audited enterprise comparison covering all four purchasing approaches with matched quality, workload and full operating costs. The absence of that comparison does not make the customer evidence worthless. It means buyers should use these cases to

EXHIBIT 15

Three ways customers changed the economics

Published customer accounts; each retains its original attribution and scope

CHANGE THE MODEL

72%

fewer Gumloop credits after switching from Opus 4.8 to GLM-5.2

Company's own July account; credits are not a complete operating-cost measure.

SPECIALIZE THE MODEL

~60%

lower inference costs for EliseAI with a fine-tuned 4B-parameter model



P90 LATENCY

2.2 s before

250 ms after

Provider-published (Baseten); 99% accuracy reported for one feature.

CHANGE WHO OPERATES IT

Managed

inference via Fireworks for Sourcegraph Cody; self-hosting judged less viable

Fireworks case (Jan. 2025), corroborated by Sourcegraph documentation.

Sources: Gumloop; Baseten; Fireworks; Sourcegraph. None is an audited, matched-quality total-cost comparison. [29, 30, 31, 32]

form hypotheses and interview questions, then verify the economics against their own work. A supplier's successful customer is a useful lead; it is not a transferable budget.

> *A supplier's successful customer is a useful lead; it is not a transferable budget.*

Choosing the operating model you can defend

The evidence supports a clear starting position. For general employee assistance, evaluate a finished application first when its capabilities fit the job. For custom workflows, compare proprietary APIs with managed open-weight inference before assuming the organization should operate GPUs. For specialized, sustained or tightly controlled workloads, put self-hosting on the shortlist and require measured performance and an explicit operating owner. These are starting points for evaluation, not exceptions-free rules.

Begin with the task and its acceptable failure rate. Test representative work against that specification, including the exceptions that make production difficult. Choose the deployment approach only after establishing which models and surrounding systems can meet the requirement. Calculate the full operating cost, test recovery and access controls, and preserve enough portability to revisit the choice when a supplier, checkpoint or price changes. Evaluation should remain part of operations after the procurement decision is made.

I would be wary of any proposal that cannot explain both why it needs the selected model and who will operate the resulting service. Buying the strongest available model for every request can waste money. Taking on infrastructure to avoid a modest API bill can do the same. The stronger proposal identifies where additional capability changes an outcome, where a smaller model is sufficient and where control is worth a premium.

Open weights make that proposal easier to construct because they expand the available suppliers, operating arrangements and continuity options. Proprietary services remain valuable when their capabilities or finished products reduce the work an enterprise must do. Leadership's responsibility is to connect those purchases to reliable business outcomes and retain the evidence for the decision. That requires an operating budget, an evaluation process and accountable people long after the launch announcement has passed.

Methodology, limitations and disclosure

This report combines public model repositories and licenses, evaluator methodologies and results, published service prices, customer accounts and enterprise research. Its evidence and price cutoff is October 29, 2026. Prices and availability can change; a quoted rate does not establish capacity, eligibility, identical service features or equal task quality. Model names refer to the specified versions, and announced weights are distinguished from files available at the cutoff.

The financial scenarios are calculations, not field measurements. No production deployment, customer interview or independent GPU benchmark was conducted for this report. Token volumes are stipulated rather than measured from a shared corpus; different tokenizers, reasoning behavior and retry rates can change actual consumption. The calculations omit caching, batch discounts and promotional credits unless explicitly stated. The illustrative budget adds assumed operating costs to demonstrate a method, not to represent a typical enterprise.

Futurum data are attributed to the published report or the supplied platform research. Raw respondent-level data were not available for this analysis. Token and spending figures from different populations are not used to calculate a savings ratio. Changes in model-family usage between waves are not treated as tracked customer migrations. Conflicting ROI findings and an unexplained incident-survey denominator were excluded from the argument pending clarification. Disputed percentages in Amplify and Notion's presentations were likewise omitted.

Customer outcomes retain their original attribution and scope. This research did not establish a complete, independently audited cost comparison across all four deployment approaches. Exact settings behind some Mistral launch comparisons remain unreconciled, and the MiMo benchmark endpoint has not been conclusively mapped to the downloadable RL variant. These limits constrain the comparisons; they do not require assuming that the underlying claims are false.

Techstrong is part of The Futurum Group. Futurum research is identified as such and considered alongside external evidence. The report's recommendations are editorial judgments based on the documented findings and explicitly labeled scenarios.

Sources

All sources checked during the research and verification process through October 29, 2026. Repository and pricing pages are live documents. Source numbers apply consistently to the main report and sidebars.

1 Joel Khalili, WIRED: Mistral Says Its AI Model Le Chonk Has Leapfrogged China in the Open-Weight Race, October 29, 2026
https://www.wired.com/story/mistral-says-le-chonk-surpasses-chinese-models-cybersecurity/

2 Artificial Analysis: Mistral Large 4 release evaluation, October 29, 2026
https://www.linkedin.com/pulse/mistral-large-4-now-open-weights-artificial-analysis-ghrpc

3 Open Source Initiative: Open Source AI Definition 1.0
https://opensource.org/ai/open-source-ai-definition

4 Mistral Large 4 official weight repository
https://huggingface.co/mistralai/Mistral-Large-4-1T-A52B/tree/main

5 Mistral Ecosystem Open-Weight License
https://huggingface.co/mistralai/Mistral-Large-4-1T-A52B/blob/main/LICENSE

6 GLM-5.3 official repository
https://huggingface.co/zai-org/GLM-5.3/tree/main

7 GLM-5.3 license
https://huggingface.co/zai-org/GLM-5.3/blob/main/LICENSE

8 DeepSeek V4.1 Flash 1017 official repository
https://huggingface.co/deepseek-ai/DeepSeek-V4.1-Flash-1017/tree/main

9 Xiaomi MiMo-V2.6-Pro-RL official repository
https://huggingface.co/XiaomiMiMo/MiMo-V2.6-Pro-RL/tree/main

10 Qwen3.5-9B official model card
https://huggingface.co/Qwen/Qwen3.5-9B

11 Kimi K3.1 Pro release and announced weight availability
https://www.kimi.com/en/blog/kimi-k3-1-pro

12 Artificial Analysis: Google has released Gemini 4 Carbon, October 28, 2026
https://www.linkedin.com/pulse/google-has-released-gemini-4-carbon-artificial-analysis-xhi4c

13 Artificial Analysis: DeepSeek V4.1 Flash 1017 evaluation
https://artificialanalysis.ai/models/deepseek-v4-1-flash-1017

14 Artificial Analysis Intelligence Index methodology
https://artificialanalysis.ai/methodology/intelligence-benchmarking

15 Artificial Analysis Coding Agent Index methodology
https://artificialanalysis.ai/methodology/coding-agents-benchmarking

16 Artificial Analysis Cyber Index methodology
https://artificialanalysis.ai/methodology/cyber-index

17 UK AI Security Institute: How far behind the frontier are leading open-weight models on cyber?
https://www.aisi.gov.uk/blog/how-far-behind-the-frontier-are-leading-open-weight-models-on-cyber

18 Nick Patience, Futurum: Enterprises Embrace Open Weights but Infrastructure Sellers Profit Most, October 2026
https://futurumgroup.com/insights/enterprises-embrace-open-weights-but-infrastructure-sellers-profit-most/

19 Amplify Partners: The 2026 AI Engineering Report
https://www.amplifypartners.com/blog-posts/the-2026-ai-engineering-report

20 Notion: AI Engineering Survey
https://www.notion.com/lp/ai-engineering-survey

21 Vercel: AI Gateway Production Index, October 2026, covering September traffic
https://vercel.com/blog/ai-gateway-production-index-october-2026

22 OpenAI Help Center: ChatGPT Business overview
https://help.openai.com/en/articles/8792828-chatgpt-business-overview

23 Anthropic: Claude subscription and API pricing
https://claude.com/pricing

24 vLLM: Conserving memory
https://docs.vllm.ai/en/latest/configuration/conserving_memory/

25 vLLM: Mistral Large 4 serving recipe
https://recipes.vllm.ai/mistralai/Mistral-Large-4-1T-A52B

26 Together AI: Model pricing
https://www.together.ai/pricing

27 Artificial Analysis: Mistral Large 4 performance and pricing
https://artificialanalysis.ai/models/mistral-large-4

28 Lambda: GPU cloud pricing
https://lambda.ai/pricing

29 Gumloop: Open-weight AI models, July 2026
https://www.gumloop.com/blog/open-weight-ai-models

30 Baseten: How EliseAI trains specialized models
https://www.baseten.co/resources/customers/how-eliseai-trains-specialized-models/

31 Fireworks: Sourcegraph code-completion customer case, January 2025
https://fireworks.ai/customers/sourcegraph-code-completion-case-study

32 Sourcegraph: Cody auto-edit documentation
https://sourcegraph.com/docs/cody/capabilities/auto-edit

33 MLCommons: MLPerf Inference Datacenter
https://mlcommons.org/benchmarks/inference-datacenter/

34 Mistral Large 4 open-weight release announcement
https://mistral.ai/news/mistral-large-4-oss/



The burden of proof belongs equally to the vendor selling a premium subscription and the team promising savings from operating its own model.

Techstrong

Techstrong is the technology media, research and events network for builders and buyers of cloud-native, DevOps, security and AI infrastructure. Techstrong is part of Futurum Media.

Futurum Research **Futurum** Media | Techstrong a Futurum company **Tech Field Day** **Techstrong TV**

OPEN WEIGHTS, REAL COSTS · A TECHSTRONG SPECIAL REPORT · OCTOBER 2026

Images in this report are AI-generated conceptual illustrations and do not depict real facilities, products or people.