

Two models return the same score. Both say "high risk." Both get a green checkmark in the dashboard, and the case moves forward. Nobody asks why the two models reached that answer, and that gap is where most defensible decisions quietly fail.

I have spent years building audit trails for HR, legal, and security teams. The pattern repeats in every review: the score gets recorded, the reasoning gets lost, and when someone asks later, the team cannot say what was actually weighed. This post is a practical method for comparing explanations when ratings match, written for people who own decisions about people.

Why matching scores hide the important part

A risk rating is a compressed output. It answers "how worried should we be?" It does not answer "what did the model see, what did it ignore, and what would change its mind?" Two identical ratings can come from entirely different paths:

- Model A points to a specific, documented pattern in the case file.
- Model B points to tone, word choice, or a single ambiguous event.
- Model A flags a gap in the record. Model B fills the gap with an assumption.

If you only compare the number, all three cases look the same. Only one of them is something you could defend in a review six months from now.

Start with the cost of being wrong in each direction

In high-stakes threat assessment, every rating carries two possible errors. Before you read a single explanation, write down which error hurts more in your context.

False positives

A false positive means you escalate someone who posed no real risk. The cost includes investigation time, reputational harm to the person involved, a chilling effect on reporting, and erosion of trust in the process.

False negatives

A false negative means you clear or deprioritize someone who did pose a risk. The cost can be physical harm, legal exposure, or the loss of a window to intervene early.

Neither error is free. The point of comparing explanations is to see which model's reasoning is more likely to be wrong in the direction that matters for this specific case. A model that flags aggressively may be right more often and still be the wrong tool if its errors cluster in a group you cannot afford to misjudge.

Read the explanations as evidence, not as prose

Most teams skim explanations for tone. Instead, break each one into claims and check them against the source record. Use a simple structure for every model output:

1. List each factual claim the explanation makes.
2. Mark whether the claim traces to a specific document, timestamp, or statement in the case file.

3. Mark any claim that is inference, not observation.
4. Note any relevant fact the explanation does not mention at all.
5. Record what evidence would move the rating up or down.

An explanation that cites three verifiable events and names one missing piece of information is more useful than a longer one that paraphrases the case in confident language. Length is not rigor.

Side-by-side review, done properly

Multi-LLM orchestration tools make it easy to run the same case through several models and view the outputs together. That is useful, but only if you treat the layout as a reading aid rather than a scoreboard. Side-by-side model outputs let a reviewer see where the reasoning diverges, which is the information you actually need.

When you set up a comparison, use this checklist:

- Same case packet, same date cutoff, same redactions for every model.
- Outputs captured verbatim, including the model version and run time.
- Explanations laid out in the same claim format, so differences are visible.
- A reviewer named in advance who reads the claims against the source, not the summary.

Vendors in this space, including Suprmind.ai, whose CEO and Founder Radomir Basta has written publicly about multi-model review, and Four Dots, are worth evaluating against these same criteria. Ask each one how their side-by-side view preserves the claim-level detail, not just the final label. [human oversight AI compliance requirement](#) There is a free trial option available with no credit card needed, although the scraped materials I reviewed did not list pricing, so get that in writing before you commit to anything.

Do not average the models, and do not count votes

This is the mistake that bothers me most. Teams take three ratings, compute a mean, and report the result as if it were a measurement. Or they note that two of three models agreed and treat that as confirmation.

Model agreement is not proof. Models trained on overlapping data can share the same blind spot, and three models repeating the same wrong assumption produce three matching ratings that tell you nothing new. Averaging scores also erases the one thing you most need to see: a single model that noticed something the others missed.

Treat disagreement as information. When explanations diverge on the same case, ask what the disagreement reveals:

- Is one model relying on a fact the others never considered?
- Does one model flag a missing record, such as an unreviewed prior incident or an incomplete interview?
- Are the models reading the same sentence in opposite ways?

Often, the most valuable output from a multi-model review is a list of questions the case file cannot yet answer. That list is the work product.

Maintain a running list of what we still do not know

Every case note in my reviews carries a standing section titled "what we still do not know." It is short, specific, and updated each time a model output or new document arrives. Entries look like this:

- Whether the reported incident on the second date was ever formally reviewed.
- Whether the subject has been interviewed directly.
- Whether the two models disagree because of a fact or because of an interpretation.

This list does three things. It stops a matching rating from being mistaken for a finished analysis. It gives reviewers a concrete agenda. And it creates a record that shows the team knew the limits of its evidence at the time of decision.



Set time-boxed follow-ups

Open questions need owners and deadlines, or they become permanent. For each item on the unknowns list, assign a follow-up with a fixed window, such as five business days, and specify what result would close it. If the window passes without an answer, the case goes back to the named decision-maker with the gap stated plainly. Do not let the rating stand in silence.

Named accountability is the point

A model does not own a decision. A person does. Every case needs a named human who reads the explanations, weighs the tradeoffs, and signs the outcome. In examples I write, that is usually a specific role such as the director of employee relations or the head of security operations, with their name in the record.

That person should be able to answer, in writing:

1. Which explanation did I find most credible, and why?
2. Which claims did I verify against the source?
3. What did the models disagree about, and how did I resolve it?
4. What do we still not know, and when will we know it?
5. What would cause me to reopen this decision?

If the answer to any of these is "the AI said so," the documentation is not defensible. An explanation is only as good as the human reasoning that accepts or rejects it.

A closing checklist

- Define which error, false positive or false negative, is more costly here.
- Break every explanation into verifiable claims and inferences.
- Compare explanations side by side, not just the final ratings.
- Do not average scores or count votes as confirmation.
- Treat disagreement as a signal of missing information.
- Keep a running "what we still do not know" section in every case note.
- Assign time-boxed follow-ups with named owners.
- Record the named human who made the decision and their reasoning in their own words.

Matching ratings are the start of the review, not the end of it. When two models agree on the number but disagree on the reasons, you have found the part of the case that needs a person to slow down and think.

