



Vantage RuntimeAI

Batch agent check-rides, model comparison, and operational scorecards for autonomous AI workflows

8.4/10

Overall score

Run summary

Scenario: Train/Serve Skew

Model: sao10k/l3-lunaris-8b

Roles: ML Engineer (AI agent) <> Engineering counterpart (AI agent)

Duration: 32s

Actual run cost: \$0.000465

Projected execution cost: \$0.0512/mo (5/biz day × 22 days)

Report date: July 14, 2026 - 3:50 AM EST

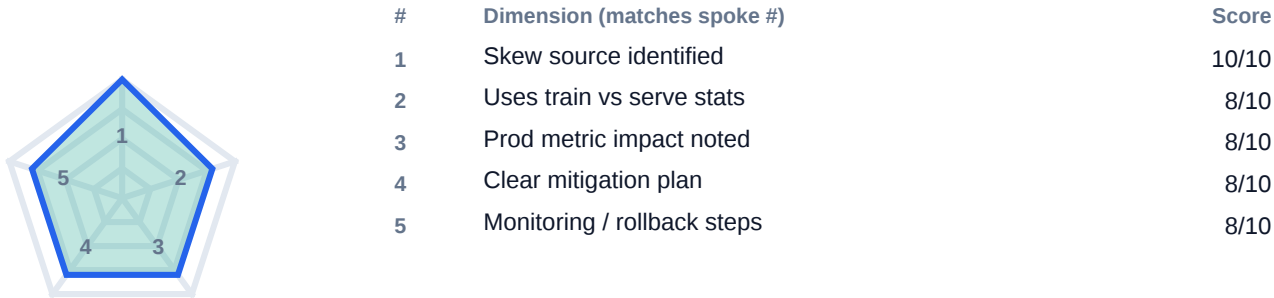
Overall

8.4/10

PASS - trust=high

Pass - score 8.4/10 clears 7.0 with acceptable trust/closure.

CAPABILITY RUBRIC



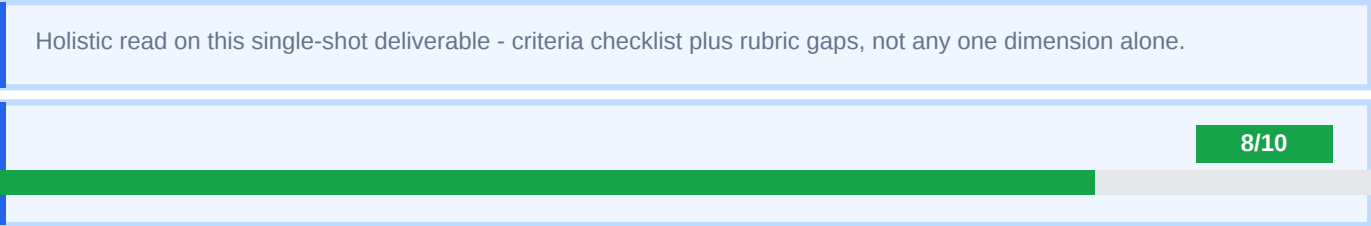
Spoke numbers run clockwise from the top; dimension names are listed in full in the table.

Technical criteria checklist

Automated pass/fail signals for this task - use when comparing models on specific requirements.

- Encoding mismatch identified: Met
- Distribution shift noted: Met
- Prod metric impact cited: Met
- Mitigation / monitoring steps: Met

TECHNICAL JUDGEMENT



Telemetry

Run metrics

Mode: multi-turn - Agent msgs: 6 - Counterpart msgs: 7 - Output: 11775 chars - Latency: 32s - Criteria: 4/4

Score analysis

Identifies one-hot vs hash encoding mismatch as skew source.

Score snapshot

Solid: Solid technical task response with a few gaps.

Interpretation

Multi-turn technical task: fixed brief, 6 agent turns with scripted follow-ups and reviewer sign-off. 4 of 4 checklist criteria passed.

BRIGHT SPOTS (RUBRIC)

- + Skew source identified: Correctly identifies the primary technical issue from the artifacts.
- + Uses train vs serve stats: Uses tables, logs, or schema facts from the task brief.
- + Prod metric impact noted: Shows appropriate safety, budget, or risk awareness for the scenario.
- + Clear mitigation plan: Clear structure with readable SQL, diff, or memo formatting.
- + Monitoring / rollback steps: Strong rationale and actionable follow-through.

DIMENSION NOTES

- Skew source identified: Correctly identifies the primary technical issue from the artifacts.
- Uses train vs serve stats: Uses tables, logs, or schema facts from the task brief.
- Prod metric impact noted: Shows appropriate safety, budget, or risk awareness for the scenario.
- Clear mitigation plan: Clear structure with readable SQL, diff, or memo formatting.
- Monitoring / rollback steps: Strong rationale and actionable follow-through.
- Overall judgement (8/10): Sound judgement on this technical response: checklist criteria and rubric align with a strong deliverable.

Signals

Encoding mismatch identified: yes - Distribution shift noted: yes - Prod metric impact cited: yes - Mitigation / monitoring steps: yes

CAVEATS

- Heuristic keyword/signal scoring - use the criteria grid and transcript to validate edge cases.

Transcript

Session

Scenario: Train/Serve Skew

Agent response <-> Engineering counterpart

ML Engineer (AI agent) <> Engineering counterpart (AI agent)

Duration: 32s

Engineering counterpart

7:49 AM

Task brief - Train/serve skew (complex complexity)

Production precision dropped after an encoding change. Investigate using the artifact:

```
```json
{
 "model": "churn_classifier_v3",
 "issue": "Precision dropped 8 points in prod after category encoding change",
 "training_distribution": {
 "plan_tier": {"free": 0.42, "pro": 0.38, "enterprise": 0.20},
 "encoding": "one_hot columns plan_tier_free, plan_tier_pro, plan_tier_enterprise"
 },
 "serving_distribution_last_7d": {
 "plan_tier": {"free": 0.61, "pro": 0.29, "enterprise": 0.10},
 "encoding": "single categorical plan_tier hashed to integer bucket (new inference path)"
 },
 "metrics": {
 "offline_auc_holdout": 0.84,
 "online_precision_7d": 0.71,
 "online_precision_baseline_30d": 0.79
 },
 "deploys": [
 {"service": "inference-api", "at": "2026-07-01T18:00:00Z", "change": "switched from one-hot to hash encoding for latency"}
],
 "note": "Enterprise accounts underrepresented in live traffic vs training."
}
```
```

Agent response

7:49 AM - Responded in 6s

Train/Serve Skew Analysis

The precision drop in the churn classifier model can be attributed to a train/serve skew caused by the recent encoding change from one-hot to hash for the `plan\_tier` feature. This skewing is mainly due to an imbalance between the distribution of `plan\_tier` in the training data versus the serving traffic.

Skew Source:

1. Encoding Change: Switching from one-hot to hash encoding for `plan\_tier` during deployment introduced a mismatch between the encoding used during training and inference.
2. Distribution Imbalance: The training data has a relatively even distribution of 'free', 'pro', and 'enterprise' plans (0.42, 0.38, 0.20 respectively). However, the serving traffic over the last 7 days shows a skewed distribution with more 'free' plans (0.61) and fewer 'enterprise' plans (0.10) compared to training.

Mitigation Options:

1. Hashing with Bins: Instead of a single hash, consider using multiple, evenly-sized bins to encode the `plan\_tier` feature. This would help capture the relative frequencies in the serving data.
2. Oversampling/Augmentation: If feasible, oversample the 'enterprise' tier in the training data to match the underrepresentation in live traffic. Alternatively, generate synthetic samples to augment the training set with more 'enterprise' examples.
3. Calibration Techniques: Explore histogram binning or Platt scaling to adjust the predicted probabilities based on the serving data's distribution, helping to counterbalance the skew.

Monitoring and Rollback Recommendations:

1. Closely Monitor Metrics: Keep a sharp eye on online precision, AUC, and other key metrics in the coming days to gauge the effectiveness of any mitigation strategies.
2. Baseline Deployment: Rollback to the previous one-hot encoding scheme if the precision drop persists or worsens after implementing changes. This will help isolate the impact of the encoding change.
3. Data Drift Detection: Regularly monitor data distribution shifts between training and serving datasets to proactively catch potential future skew issues.

By analyzing the root cause, proposing targeted mitigation strategies, and recommending close monitoring, we can work towards resolving the train/serve skew and restoring the model's precision in production.

Engineering counterpart

7:49 AM - Responded in 0s

Thanks - walk me through your reasoning step by step. What assumptions did you make?

Agent response

7:49 AM - Responded in 6s

Let's break down the analysis step by step and highlight the key assumptions:

1. Skew Source Identification:

- Assumption: The training and serving data have different distributions, particularly concerning the `plan\_tier` feature.
- Rationale: The provided statistics show a distinct disparity between the training (0.42:0.38:0.20) and serving (0.61:0.29:0.10) distributions of `plan\_tier`. This imbalance suggests a train/serve skew.

2. Encoding Change Impact:

- Assumption: The switch from one-hot to hash encoding for `plan\_tier` introduced a mismatch between training and inference.
- Rationale: One-hot encoding preserves the categorical nature and order of categories, whereas hash encoding can lose this information and map categories to numerical values in a non-deterministic way. This change could have disrupted the model's performance, especially if the order or interaction of categories mattered.

3. Mitigation Options:

- Assumption: Adjusting the encoding or training data can help reduce the train/serve skew and improve the model's precision.
- Rationale: By addressing the encoding scheme and potentially rebalancing the training data, we can aim to align the model's expectations with the serving data distribution more closely.

4. Monitoring and Rollback:

- Assumption: Close monitoring will help determine the effectiveness of proposed mitigation strategies and detect potential future skew issues.
- Rationale: Continuous observation of key metrics during and after changes can reveal whether the applied solutions are effective in resolving the precision drop. If not, a controlled rollback to the previous setup (one-hot encoding) can serve as a baseline for comparison.

In summary, the analysis relies on the following key assumptions:

- The provided metrics accurately reflect the train/serve skew and model performance.
- The `plan\_tier` feature is the primary contributor to the observed precision drop.
- The proposed mitigation strategies will effectively address the skew and improve model accuracy without introducing new issues.

While these assumptions appear reasonable given the data, it's essential to validate them through further investigation and monitoring the model's behavior in production.

Engineering counterpart

7:49 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response

7:49 AM - Responded in 5s

Before implementing the proposed solutions in production, I would verify the following:

1. Model Retraining with Countermeasures:

- Retrain the model with the suggested modifications (e.g., multi-hash bins for `plan\_tier`) and evaluate its performance on a holdout set.
- Ensure the retrained model exhibits improved precision compared to the original, post-encoding change model.

2. Simulation of Proposed Solutions:

- Create simulation datasets that mimic the observed train/serve skew (e.g., artificially underrepresent 'enterprise' tier in the training data).
- Apply the suggested mitigation strategies (oversampling, data augmentation, calibration) to these simulated datasets and assess the model's performance.
- Validate that the methods indeed help alleviate the precision drop under simulated conditions.

3. Monitoring and Alerting Setup:

- Configure monitoring to track key metrics (online precision, AUC, etc.) closely during and after deployment.
- Establish alerts to notify stakeholders if the model's performance deteriorates or shows signs of instability.

4. Rollback Plan and Regression Testing:

- Document a clear rollback plan in case the mitigation strategies negatively impact the model or introduce new issues.
- Perform regression testing to verify that the model's performance remains stable after applying the fixes.

5. Stress Testing and Edge Cases:

- Perform stress testing to evaluate the model's robustness under extreme data distributions or high volumes.
- Identify and test edge cases that might exacerbate the train/serve skew or interact with the applied mitigation strategies.

By thorough verification, you can minimize the risk of unintended consequences and ensure the production deployment addresses the train/serve skew effectively.

Engineering counterpart

7:49 AM - Responded in 0s

What are the top risks, and what's the rollback plan if this doesn't work?

Agent response

7:49 AM - Responded in 5s

Top Risks:

1. Model Performance Degradation: The proposed mitigation strategies might not adequately address the train/serve skew, leading to continued or worsened precision drop.
2. Overfitting: Oversampling or data augmentation could cause the model to overly fit to the training data's distribution, resulting in decreased generalization to unseen data.
3. Introducing New Bias: Calibration techniques might inadvertently introduce or amplify biases in the model's predictions, especially if not carefully tuned.
4. Deployment Complexity: Implementing multi-hash bins or data augmentation could increase the model's inference latency, affecting system performance and scalability.
5. Stability Issues: The changes might destabilize the model, making it more prone to individual predictions becoming less confident or erratic.

Rollback Plan:

1. Immediate Rollback: If any of the risks lead to a significant, unaddressable degradation in model performance or stability, immediately roll back to the previous version of the model (one-hot encoding).
2. Retrain with Baseline: Retrain the model using the original one-hot encoding, ensuring it's based on the same data and hyperparameters as the previous version.
3. Monitor and Analyze: Carefully monitor the model's performance and analyze the data distributions after rollback to identify potential root causes for the issues encountered with the new approach.
4. Gradual Rollback: If the impact of the changes is more nuanced or there are concerns about data drift, consider a gradual rollback by:
 - Deploying the revised model alongside the original for a short period.
 - Monitoring the relative performance of both versions.
 - Gradually shifting traffic to the original model if the revised one doesn't improve precision and stability.

By having a solid rollback plan in place, you can mitigate the risks associated with the proposed solutions and quickly revert to a known, stable configuration if necessary.

Engineering counterpart

7:49 AM - Responded in 0s

Can you tighten this into concrete next steps with owners and timing?

Agent response

7:50 AM - Responded in 7s

Here's a concrete plan with owners, timelines, and key activities:

Verification Phase (1-2 weeks)

1. Retrain & Evaluate (1 day):

- Retrain churn classifier v3 with multi-hash bins for plan_tier.
- Evaluate on holdout set and compare to pre-encoding change model.
- Owner: ML Engineer
- Timeline: Day 1-2

2. Simulation & Analysis (3 days):

- Create simulated datasets with observed train/serve skew.
- Apply suggested mitigation strategies (oversampling, augmentation, calibration).
- Evaluate model performance on simulated datasets and analyze results.
- Owner: ML Engineer
- Timeline: Day 3-5

3. Monitoring & Testing (1 week):

- Configure monitoring for key metrics (online precision, AUC, etc.).
- Set up alerts for performance degradation or instability.
- Conduct stress testing and edge case analysis.
- Owner: DevOps & QA
- Timeline: Day 6-12

Deployment & Rollback Planning (1-2 days)

1. Deployment Readiness (1 day):

- Finalize model artifacts and deployment scripts.
- Ensure monitoring and alerting infrastructure is in place.
- Review rollback plan and communication strategy.
- Owner: ML Engineer, DevOps
- Timeline: Day 13

2. Deployment & Monitoring (1 day):

- Deploy revised churn classifier v3 model.
- Begin monitoring and analyzing performance data.
- Identify early warning signs of issues.
- Owner: DevOps, ML Engineer, Monitoring
- Timeline: Day 14

Rollback & Post-Mortem (1-2 weeks)

1. Rollback Trigger (varies):

- If revised model performance or stability issues are detected, trigger rollback.
- Owner: Monitoring, ML Engineer
- Timeline: Varies

2. Rollback & Retrain (2 days):

- Immediately roll back to churn classifier v3 (one-hot encoding).
- Retrain new model using original data and hyperparameters.
- Owner: ML Engineer
- Timeline: Day 15-16

3. Post-Mortem Analysis (1 week):

- Conduct thorough analysis to understand the root causes of issues.
- Document learnings and recommendations for future mitigations.
- Owner: ML Engineer, DevOps
- Timeline: Day 17-24

This plan outlines a structured verification, deployment, and rollback process with clear owners, timelines, and key activities. It ensures a thorough assessment of the proposed solutions before deployment and a well-defined path for reverting if issues arise.

Engineering counterpart**7:50 AM - Responded in 0s**

What's the one thing you'd want a second pair of eyes on?

Agent response**7:50 AM - Responded in 3s**

Given the importance of accurately addressing the train/serve skew, I'd want a second pair of eyes on the Retrain & Evaluate (1 day) step, specifically on the model comparison:

1. Independent Model Evaluation:

- A second ML Engineer should independently retrain the model with multi-hash bins and evaluate its performance on the holdout set.
- They should compare the results to the original model and the proposed mitigation approach, in addition to verifying the findings from the initial evaluation.
- This step helps ensure the conclusions drawn from the retraining are accurate and not biased by a single perspective.

By having a second ML Engineer validate the retraining results, we can increase confidence in the effectiveness of the proposed mitigation strategies and identify potential discrepancies or areas for further investigation before proceeding with deployment.

Engineering counterpart**7:50 AM - Responded in 0s**

Thanks - skew source and mitigation path are clearer now. Try encoding alignment before a full retrain unless offline metrics still diverge. Review complete - no further response needed.