



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: meta-llama/llama-4-maverick

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

Duration: 2m 24s

Actual run cost: \$0.003480

Projected execution cost: \$0.3828/mo (5/biz day × 22 days)

Report date: July 14, 2026 - 3:47 AM EST

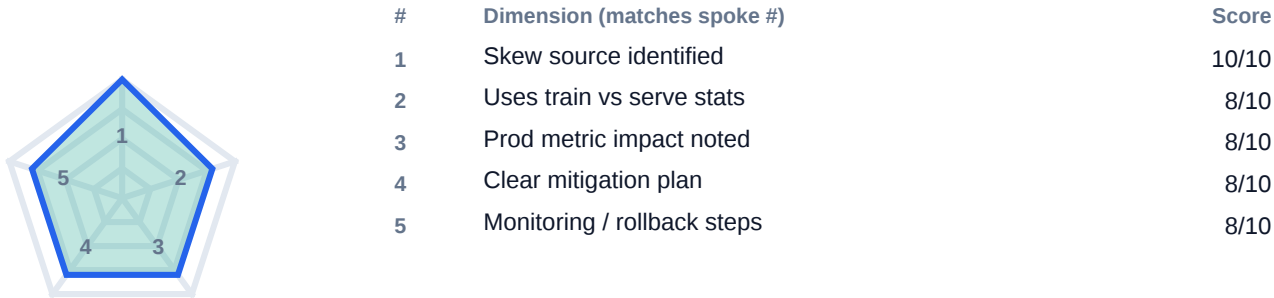
Overall

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

Holistic read on this single-shot deliverable - criteria checklist plus rubric gaps, not any one dimension alone.

8/10

Telemetry

Run metrics

Mode: multi-turn - Agent msgs: 6 - Counterpart msgs: 7 - Output: 17395 chars - Latency: 144s - 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: 2m 24s

Engineering counterpart

7:44 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

7:45 AM - Responded in 27s

Investigation of Train/Serve Skew

Step 1: Understanding the Problem

The precision of the `churn\_classifier\_v3` model dropped by 8 points in production after a change in category encoding for the `plan\_tier` feature. The task is to identify the source of the skew, suggest mitigation options, and recommend monitoring or rollback strategies.

Step 2: Analyzing the Artifact

The provided JSON artifact contains crucial information about the model, the issue, training and serving distributions, metrics, and deployment history.

- Model and Issue: The model is `churn\_classifier\_v3`, and the issue is a drop in precision after changing the encoding for `plan\_tier`.
- Training Distribution:
 - `plan\_tier` distribution: `free` (0.42), `pro` (0.38), `enterprise` (0.20).
 - Encoding: One-hot encoding into `plan\_tier\_free`, `plan\_tier\_pro`, `plan\_tier\_enterprise`.
- Serving Distribution (Last 7 days):
 - `plan\_tier` distribution: `free` (0.61), `pro` (0.29), `enterprise` (0.10).
 - Encoding: Single categorical `plan\_tier` hashed to an integer bucket.
- Metrics:
 - Offline AUC on holdout set: 0.84.
 - Online precision (last 7 days): 0.71.
 - Online precision baseline (last 30 days): 0.79.
- Deploys:
 - A change was made to the `inference-api` on `2026-07-01T18:00:00Z` to switch from one-hot to hash encoding for latency reasons.

Step 3: Identifying the Source of Skew

1. Distribution Shift: There's a noticeable difference in the `plan\_tier` distribution between training and serving data. The serving data has a higher proportion of `free` plans and a lower proportion of `enterprise` plans compared to the training data.
2. Encoding Change: The encoding method for `plan\_tier` was changed from one-hot encoding during training to hash encoding during serving. This change could potentially cause issues if the model is not robust to the different encoding schemes, especially since hash encoding can lead to collisions.

Step 4: Mitigation Options

1. Revert Encoding Change: Reverting to one-hot encoding in the serving path could mitigate the precision drop if the model's performance is significantly affected by the encoding change. However, this might impact latency.
2. Update Model Training: Retrain the model using the same hash encoding for `plan\_tier` as used in serving. This would ensure consistency between training and serving.
3. Data Augmentation or Re-sampling: Adjust the training data distribution to better match the serving distribution, focusing on increasing the representation of `free` plans and potentially decreasing `enterprise` plans, though the latter should be done cautiously to avoid losing valuable information.
4. Feature Engineering: Explore alternative encoding strategies or features that could be more robust and less sensitive to distribution shifts.

Step 5: Monitoring and Rollback Recommendations

1. Continuous Monitoring: Closely monitor the model's performance (precision, AUC, etc.) and data distributions (training and serving) post-mitigation.
2. Automated Rollback: Set up automated rollback mechanisms to quickly revert to a previous known good state (e.g., previous encoding scheme) if the precision does not recover or further degrades.
3. A/B Testing: Consider A/B testing different mitigation strategies (e.g., different encoding schemes, updated models) to determine the most effective approach without fully committing to it.
4. Logging and Alerting: Ensure that detailed logs are kept for serving data distributions and model predictions, and set up alerts for significant deviations in monitored

Engineering counterpart
7:45 AM - Responded in 0s

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

Agent response

7:45 AM - Responded in 27s

Step-by-Step Reasoning

1. Understanding the Problem:

- The task is to investigate a drop in precision of a classification model (`churn\_classifier\_v3`) after a change in the production environment.
- The change involved switching from one-hot encoding to hash encoding for the `plan\_tier` feature.

2. Analyzing Distributions:

- The training data has a different distribution of `plan\_tier` compared to the serving data. Specifically, `free` plans are more common, and `enterprise` plans are less common in serving data.
- Assumption: The difference in distribution could affect model performance if the model is not robust to such shifts.

3. Examining Encoding Changes:

- The encoding for `plan\_tier` changed from one-hot to hash encoding.
- Assumption: This change could impact model performance because the model was trained on one-hot encoded data but is being served with hash encoded data. Hash encoding can lead to collisions, potentially affecting the model's ability to distinguish between categories.

4. Reviewing Metrics:

- The model's offline AUC is 0.84, indicating good performance on the holdout set.
- The online precision dropped from 0.79 (baseline over the last 30 days) to 0.71 (over the last 7 days) after the encoding change.
- Assumption: The drop in precision is related to the changes made (either the encoding change, the distribution shift, or both).

5. Identifying Skew Sources:

- Distribution Shift: The change in `plan\_tier` distribution between training and serving data is a potential source of skew. Models can be sensitive to the distribution of input features.
- Encoding Change: The switch from one-hot to hash encoding is another potential source. The model's performance could degrade if it relies on the specific encoding used during training.

6. Mitigation Strategies:

- Reverting Encoding Change: If the hash encoding is the primary cause, reverting to one-hot encoding could help. This assumes that one-hot encoding is feasible in production despite potential latency concerns.
- Retraining the Model: Training the model with hash encoding could align the training and serving conditions, potentially improving precision. This assumes that retraining is feasible and that the model can adapt to hash encoding.
- Adjusting Training Data Distribution: Making the training data distribution more similar to the serving data distribution could help if distribution shift is a significant issue. This assumes that such adjustments are feasible and won't negatively impact the model's performance on other data distributions.

7. Monitoring and Rollback:

- Continuous monitoring is essential to catch any further performance degradation or improvement.
- Automated rollback can mitigate risks by quickly reverting to a known good state if needed.
- Assumption: Having a robust monitoring and rollback system in place is crucial for managing model performance in production.

Assumptions Made

1. Model Sensitivity: The model is sensitive to both the encoding scheme and the distribution of `plan\_tier`.
2. Impact of Encoding: The change from one-hot to hash encoding has a significant impact on the model's performance.
3. Distribution Shift Impact: The shift in `plan\_tier` distribution affects the model's performance.
4. Feasibility of Mitigations: The proposed mitigation strategies (reverting encoding, retraining, adjusting training data) are technically feasible and can be implemented without significant additional issues.

5. Monitoring and Rollback Capabilities: The production environment is capable of supporting detailed monitoring and automated rollback.

These assumptions are based on the information provided and general knowledge about machine learning model deployment and management. In a real-world scenario, further investigation and testing would be necessary to validate these assumptions.

What would you verify before shipping this to production?

Agent response

7:46 AM - Responded in 35s

Pre-Production Verification Steps

Before shipping the recommended mitigations to production, several verification steps should be taken to ensure the solutions are effective and do not introduce new issues.

For Reverting Encoding Change:

1. Latency Impact Assessment: Verify that reverting to one-hot encoding does not significantly impact the latency of the inference API. If latency is a concern, assess whether optimizations can be made to mitigate any increase.
2. Precision Recovery: Test the model with one-hot encoding in a controlled environment to confirm that precision recovers to acceptable levels (close to or better than the baseline of 0.79).
3. Integration Testing: Perform thorough integration testing to ensure that reverting to one-hot encoding does not break any downstream processes or introduce inconsistencies.

For Retraining the Model with Hash Encoding:

1. Model Performance: Retrain the model using hash encoding for `plan\_tier` and evaluate its performance on a holdout dataset. Verify that the model's precision (and other relevant metrics) is satisfactory.
2. Hyperparameter Tuning: If necessary, perform hyperparameter tuning to ensure the retrained model is optimized for the hash encoding scheme.
3. Comparison with Original Model: Compare the performance of the retrained model with the original model to understand any differences and ensure the retrained model is at least as good, if not better.

For Adjusting Training Data Distribution:

1. Data Preparation: Prepare a new training dataset that more closely matches the serving data distribution for `plan\_tier`.
2. Model Retraining and Evaluation: Retrain the model on this new dataset and evaluate its performance. Check if the model's precision improves on the serving data distribution.
3. Generalization Check: Verify that adjusting the training data distribution does not negatively impact the model's performance on other important data distributions or scenarios.

General Verification Steps:

1. A/B Testing: Consider conducting A/B testing in a production-like environment to compare the performance of different mitigation strategies (if multiple are being considered).
2. Monitoring Setup: Ensure that monitoring is in place to track the model's performance, data distributions, and other relevant metrics post-deployment.
3. Rollback Planning: Confirm that rollback mechanisms are in place and tested, so it's possible to quickly revert to a previous state if issues arise.
4. Stakeholder Communication: Communicate the planned changes, expected outcomes, and any potential risks to relevant stakeholders.

By taking these verification steps, you can increase confidence that the chosen mitigation strategy will be effective in production and minimize the risk of introducing new problems.

Engineering counterpart
7:46 AM - Responded in 0s

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

Agent response

7:46 AM - Responded in 19s

Top Risks

1. Precision Doesn't Recover: The mitigation strategies might not fully recover the lost precision, or they might not be as effective as anticipated.
2. Latency or Performance Issues: Reverting to one-hot encoding or implementing other changes could potentially introduce latency or performance issues, especially if not properly optimized.
3. Model Overfitting or Underfitting: Adjusting the training data distribution or retraining the model with hash encoding could lead to overfitting or underfitting if not managed correctly.
4. Data Quality or Availability Issues: Changes in data preparation or distribution might uncover or exacerbate existing data quality issues.
5. Downstream System Compatibility: Changes to the encoding or model output could potentially affect downstream systems or processes that rely on the model's output.

Rollback Plan

1. Revert to Previous Model or Encoding: If the precision doesn't recover or if other issues arise, the first step would be to revert to the previous model version or encoding scheme that was known to work.
2. Automated Rollback: Implement automated rollback mechanisms that can quickly switch back to a previous known good state. This could involve having multiple model versions or configurations readily available for deployment.
3. Monitoring and Alerting: Ensure that there is robust monitoring and alerting in place to quickly identify if the mitigation strategies are not working as expected. This allows for a timely decision to roll back.
4. Fallback to Baseline: If the new strategies are not effective, fall back to the baseline configuration (previous model and encoding scheme) until further adjustments or alternative solutions can be tested and validated.
5. Contingency Planning for Alternative Solutions: Have a plan in place for exploring alternative mitigation strategies or solutions if the initial approaches do not work. This could involve further analysis, model retraining with different techniques, or exploring different feature engineering strategies.

Steps for Rollback

1. Identify Trigger: Determine the metrics or conditions that will trigger a rollback (e.g., precision below a certain threshold, significant increase in latency).
2. Automate Rollback Process: Set up an automated process to roll back to a previous version. This could involve scripts or tools that can quickly deploy a previous model or configuration.
3. Test Rollback Process: Before deploying the mitigation strategies, test the rollback process to ensure it works as expected and can be executed quickly.
4. Communicate with Stakeholders: Inform stakeholders about the rollback plan and the conditions under which it will be triggered, ensuring transparency and readiness for any potential issues.

By having a well-planned rollback strategy, you can mitigate the risks associated with deploying changes to a production model and ensure a quick recovery if issues arise.

Engineering counterpart
7:46 AM - Responded in 0s

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

Agent response

7:47 AM - Responded in 27s

Concrete Next Steps with Owners and Timing

Immediate Next Steps (Within 48 hours)

1. Revert Encoding Change:
 - Task: Assess the feasibility of reverting to one-hot encoding in the serving path.
 - Owner: Engineering Team
 - Timing: Complete assessment within 24 hours.
2. Retrain Model with Hash Encoding:
 - Task: Initiate retraining of the model using hash encoding for `plan\_tier`.
 - Owner: Data Science Team
 - Timing: Start retraining within 48 hours.
3. Adjust Training Data Distribution:
 - Task: Prepare a plan to adjust the training data distribution to more closely match the serving data distribution.
 - Owner: Data Engineering/Data Science Team
 - Timing: Complete planning within 48 hours.

Short-Term Next Steps (Within 1 week)

1. Testing and Validation:
 - Task: Test and validate the performance of the retrained model (with hash encoding) and the impact of reverting to one-hot encoding.
 - Owner: Data Science/Engineering Team
 - Timing: Complete testing and validation within 5 days.
2. A/B Testing:
 - Task: Plan and initiate A/B testing for the different mitigation strategies (if applicable).
 - Owner: Data Science/Engineering Team
 - Timing: Start A/B testing within 5-7 days.
3. Rollback Plan Finalization:
 - Task: Finalize the rollback plan, including automated rollback mechanisms and monitoring/alerting setup.
 - Owner: Engineering/DevOps Team
 - Timing: Complete within 7 days.

Mid-Term Next Steps (Within 2 weeks)

1. Deployment of Chosen Mitigation:
 - Task: Deploy the chosen mitigation strategy (based on testing and validation results) to production.
 - Owner: Engineering/DevOps Team
 - Timing: Deploy within 10-14 days.
2. Monitoring and Evaluation:
 - Task: Continuously monitor the model's performance and other relevant metrics post-deployment.
 - Owner: Data Science/Engineering Team
 - Timing: Ongoing.

Key Milestones and Checkpoints

- Day 3: Review the assessment of reverting to one-hot encoding and the status of model retraining.
- Day 7: Review the results of testing and validation for the mitigation strategies and finalize the A/B testing plan.

- Day 10: Deploy the chosen mitigation strategy to production.
- Day 14: Review post-deployment performance and adjust as necessary.

By outlining specific tasks, owners, and timelines, we can ensure that the mitigation strategies are thoroughly tested and validated before deployment, and that there is a clear plan in place for monitoring and potential rollback.

Engineering counterpart**7:47 AM - Responded in 0s**

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

Agent response**7:47 AM - Responded in 9s**

One Thing for a Second Pair of Eyes

The one thing I'd want a second pair of eyes on is the Data Distribution Adjustment Plan.

Adjusting the training data distribution to more closely match the serving data distribution is a critical step, and it's essential to ensure that this adjustment is done correctly. A second pair of eyes can help validate the plan by:

1. Reviewing the methodology: Ensuring that the approach to adjusting the data distribution is statistically sound and won't introduce any biases.
2. Checking for potential issues: Identifying potential issues with the adjusted data distribution, such as over-representation of certain classes or features.
3. Verifying data quality: Ensuring that the adjusted data distribution doesn't compromise data quality or introduce new data quality issues.

Having a second pair of eyes review the Data Distribution Adjustment Plan can help ensure that the plan is robust, effective, and doesn't introduce new risks or issues. This review can be done by a colleague or peer with expertise in data preprocessing, data quality, or machine learning.

Engineering counterpart**7:47 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.