



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: qwen/qwen-2.5-7b-instruct

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

Duration: 1m 21s

Actual run cost: \$0.001320

Projected execution cost: \$0.1452/mo (5/biz day × 22 days)

Report date: July 14, 2026 - 3:48 AM EST

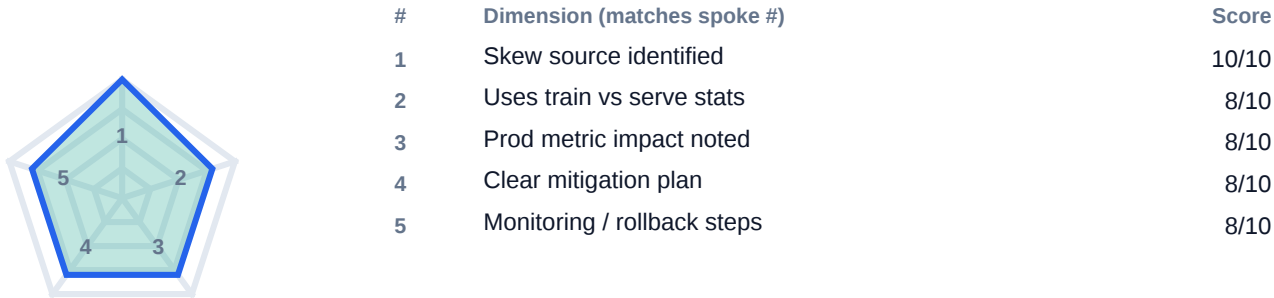
Overall

8.4/10

PASS - trust=high

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

CAPABILITY RUBRIC



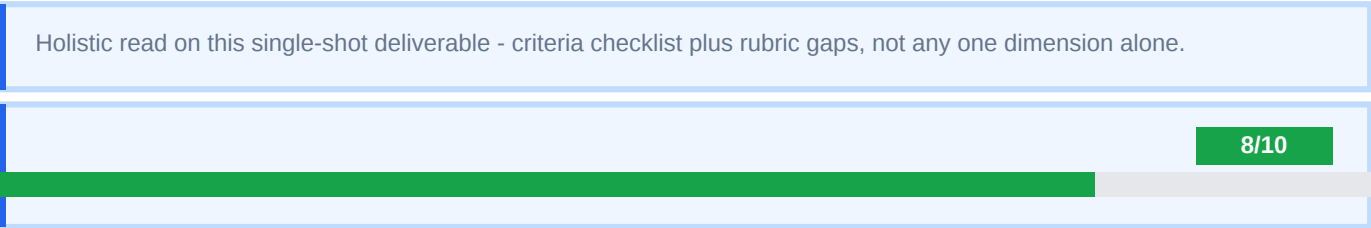
Spoke numbers run clockwise from the top; dimension names are listed in full in the table.

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: 21750 chars - Latency: 81s - 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: 1m 21s

Engineering counterpart

7:47 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

7:48 AM - Responded in 44s

Skew Source Investigation

The issue appears to be a result of data skew between the training and serving distributions, specifically in how the `plan\_tier` feature is encoded. Let's break down the evidence and analyze the sources of skew:

Training Distribution

- Plan Tier Distribution:

- Free: 42%
- Pro: 38%
- Enterprise: 20%

Serving Distribution (Last 7 Days)

- Plan Tier Distribution:

- Free: 61%
- Pro: 29%
- Enterprise: 10%

The major discrepancy is in the enterprise accounts. In the training data, enterprise accounts make up 20% of the data, whereas in the serving data, they only account for 10%. This imbalance is likely causing the model to underperform on enterprise accounts during inference.

Mitigation Options

1. Weighted Loss Function:

- Use a weighted loss function that gives more importance to the less frequent class (Enterprise accounts).
- Example: In a binary classification problem, if the positive class is Enterprise, the loss function can be adjusted as:

$$\text{Loss} = -(\alpha \times \log(p) + (1-\alpha) \times \log(1-p))$$

where α is the weight assigned to the positive class (Enterprise).

2. Synthetic Data Generation:

- Generate synthetic data points for the less frequent class (Enterprise accounts) to balance the training dataset.
- Techniques like SMOTE (Synthetic Minority Over-sampling Technique) can be used.

3. Model Retraining:

- Re-train the model on a more representative dataset that includes a better balance of enterprise accounts.
- Split the training data to ensure that the model is trained on a data distribution that closely mirrors the serving distribution.

4. Feature Engineering:

- Engineer additional features that might capture the characteristics of enterprise accounts more effectively, such as average spend or other billing-related features.

Monitoring and Rollback Recommendations

1. Monitoring:

- Set up a monitoring system that tracks the distribution of `plan\_tier` in the serving environment over time.
- Use drift detection tools to monitor changes in the feature distribution.
- Implement alerting when significant drift is detected.

2. Rollback:

- If the precision drop continues or worsens, consider rolling back to the previous model version that used one-hot

encoding.

- Perform an A/B test to compare the performance of the original and the newer model versions.

3. Continuous Learning:

- Implement a continuous learning approach where the model is retrained periodically with new data.
- Use online learning techniques to update the model incrementally.

Implementation Plan

1. Immediate Steps:

- Roll back to the previous version of the model that uses one-hot encoding.
- Monitor the precision and other relevant metrics for a few days.

2. Long-term Steps:

- Implement the weighted loss function or synthetic data generation to mitigate the skew.
- Retrain the model with a balanced dataset.
- Set up monitoring and continuous learning pipelines.

Summary

The primary source of skew is the underrepresentation of enterprise accounts in the serving data compared to the training data. Mitigating this skew through weighted loss functions, synthetic data generation, or model retraining will help improve model performance in production. Continuous monitoring and alerting are crucial to detect and address any future skew.

Would you like to proceed with any specific mitigation step or need further details on the implementation?

Engineering counterpart
7:48 AM - Responded in 0s

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

Agent response

7:48 AM - Responded in 7s

Sure, let's walk through the reasoning step by step and clarify the assumptions made.

Step-by-Step Reasoning

1. Identify the Skew

- Training Distribution:
 - Free: 42%
 - Pro: 38%
 - Enterprise: 20%
- Serving Distribution (Last 7 Days):
 - Free: 61%
 - Pro: 29%
 - Enterprise: 10%

From this, we can see that the distribution of `plan\_tier` in the serving environment is significantly different from the training data, particularly for the `enterprise` tier.

2. Assumptions

- Assumption 1: The model's performance is sensitive to the distribution of the `plan\_tier` feature.
- Assumption 2: The imbalance in the `enterprise` tier in the serving data is causing the model to underperform on this segment.
- Assumption 3: The one-hot encoding method used in training is less prone to data skew compared to the new hash encoding method.

3. Analyze the Impact of Skew

- Implication:
 - The `enterprise` tier, which is underrepresented in the serving data, is less represented in the training data.
 - This imbalance can cause the model to be less accurate on enterprise accounts when deployed in production.

4. Mitigation Options

1. Weighted Loss Function

- Reasoning:
 - The model might be biased towards predicting the majority class (Free and Pro).
 - By giving more weight to the less frequent class (Enterprise), the model can learn to handle the minority class better.
- Example:
 - If the positive class is `enterprise`, the loss function can be adjusted as:
$$\text{Loss} = -(\alpha \times \log(p) + (1-\alpha) \times \log(1-p))$$
where α is the weight assigned to the positive class (Enterprise).

2. Synthetic Data Generation

- Reasoning:
 - Generate synthetic data points to balance the training dataset.
 - Techniques like SMOTE can be used to create more synthetic examples of the underrepresented class (Enterprise).
- Example:
 - Use SMOTE to create new synthetic data points for the `enterprise` tier to match the serving distribution.

3. Model Retraining

- Reasoning:
 - Retrain the model on a dataset that better represents the serving distribution.

- This ensures that the model is trained on a more balanced dataset.
- Implementation:
 - Split the training data to ensure a better balance of `plan\_tier` classes.

4. Feature Engineering

- Reasoning:
 - Engineer additional features that capture the characteristics of `enterprise` accounts more effectively.
- Example:
 - Include features like average spend, billing patterns, or other relevant billing-related features.

Monitoring and Rollback

1. Monitoring

- Reasoning:
 - Continuous monitoring helps in detecting any drift in the feature distribution.
 - Alerting systems can be set up to notify relevant teams when drift is detected.
- Implementation:
 - Use drift detection tools like Feature Store tools to monitor changes in feature distributions.
 - Set up alerting on significant changes in the `plan\_tier` distribution.

2. Rollback

- Reasoning:
 - If the precision drop continues or worsens, revert to the previous model version.
 - Perform an A/B test to compare performance between the old and new models.
- Implementation:
 - Roll back to the previous version of the model that used one-hot encoding.
 - Set

Engineering counterpart
7:48 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response

7:48 AM - Responded in 7s

Before shipping the mitigation measures to production, it is crucial to verify several aspects to ensure the changes are effective and do not introduce new issues. Here are the key steps and verifications you should perform:

1. Reproduce and Validate the Issue Locally
2. Test the Mitigation on a Subset of Data
3. Evaluate Model Performance on Holdout Data
4. Monitor Model Performance in Staging Environment
5. Set Up Comprehensive Monitoring in Production
6. Document and Communicate Changes and Outcomes

1. Reproduce and Validate the Issue Locally

- Verify the Skew: Ensure the skew exists and is causing the precision drop.
- Reproduce the Drop: Create a local environment that mimics the production setup and verify the precision drop.

2. Test the Mitigation on a Subset of Data

- Apply Mitigation: Implement the chosen mitigation strategy (e.g., weighted loss function, synthetic data generation, retraining).
- Evaluate on Holdout Data: Test the model on a holdout dataset to ensure the mitigation improves the model's performance.

3. Evaluate Model Performance on Holdout Data

- Check Precision: Ensure the precision on the holdout data improves, particularly for the `enterprise` tier.
- Monitor Other Metrics: Ensure other metrics like AUC, recall, and F1-score are maintained or improved.

4. Monitor Model Performance in Staging Environment

- Deploy to Staging: Deploy the mitigated model to a staging environment.
- Monitor for Stability: Continuously monitor the model's performance and feature distribution in the staging environment.
- Collect Feedback: Gather feedback from any telemetry or alerts to ensure the model is behaving as expected.

5. Set Up Comprehensive Monitoring in Production

- Feature Distribution Monitoring: Set up monitoring to track the distribution of `plan\_tier` in the serving data.
- Model Performance Monitoring: Monitor metrics like precision, recall, AUC, and F1-score in production.
- Drift Detection: Use tools like MLflow, H2O.ai Flow, or custom scripts to detect any drift in feature distributions or model performance.

6. Document and Communicate Changes and Outcomes

- Document Changes: Record the changes made, the rationale, and the expected outcomes.
- Communicate Outcomes: Inform relevant stakeholders (e.g., engineering, product, and data science teams) about the changes and preliminary results.
- Prepare Rollback Plan: Ensure there is a clear plan in place for rolling back to the previous model if issues arise.

Example Implementation Plan

1. Reproduce the Issue Locally

- Local Data Setup: Use a local copy of the production data.
- Simulation: Simulate the production environment (e.g., using the same feature engineering and model deployment process).

- Verify Skew: Confirm the `enterprise` tier is underrepresented in the production data.

2. Test the Mitigation on a Subset of Data

- Weighted Loss Function:
 - Adjust the loss function in the training process.
 - Train the model with the new loss function and evaluate on a holdout dataset.
- Synthetic Data Generation:
 - Generate synthetic data for the `enterprise` tier.
 - Train the model on the augmented dataset and evaluate on the holdout dataset.
- Retraining:
 - Retrain the model on a dataset that better represents the serving distribution.
 - Evaluate the model on the holdout dataset.

3. Evaluate Model Performance on Holdout Data

- Precision: Check if the precision for the `enterprise` tier improves.
- Other Metrics: Ensure other metrics like AUC and F1-score are maintained or improved.

4. Monitor Model Performance in Staging Environment

- Deployment: Deploy the mitigated

Engineering counterpart
7:48 AM - Responded in 0s

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

Agent response

7:48 AM - Responded in 8s

Top Risks and Rollback Plan

Top Risks

1. Model Performance Degrades Further: The mitigation might not solve the precision drop, and the model performance could degrade further.
2. Introducing New Biases: The mitigation might introduce new biases or skew in the model.
3. Technical Issues: There could be technical issues during deployment or monitoring that affect the model's performance.
4. User Impact: If the model underperforms, it could negatively impact user experience or business outcomes.

Rollback Plan

1. Immediate Rollback
2. Detailed Analysis and Troubleshooting
3. Communication and Stakeholder Updates

Immediate Rollback

1. Roll Back to Previous Model:
 - Action: Roll back to the previous version of the model that used one-hot encoding.
 - Steps:
 - Deploy the previous model to the production environment.
 - Monitor the model's performance to ensure it is stable and meets the desired performance metrics.
2. Monitor Post-Rollback:
 - Steps:
 - Continuously monitor the model's performance metrics (e.g., precision, AUC) in the production environment.
 - Verify that the precision is restored to the baseline and other metrics are stable.

Detailed Analysis and Troubleshooting

1. Analyze Metrics:
 - Steps:
 - Compare the pre- and post-rollback performance metrics to identify any anomalies.
 - Check the feature distribution in the serving environment to ensure it aligns with expectations.
2. Review Logs and Alerts:
 - Steps:
 - Review logs and any alerts that were triggered during the deployment to identify any technical issues.
 - Investigate any errors or warnings that might indicate problems with the new model.
3. Examine Feature Influence:
 - Steps:
 - Analyze the feature importance and influence in the new model to understand how the changes affect the model.
 - Use techniques like partial dependence plots to understand the impact of the `plan\_tier` feature.

Communication and Stakeholder Updates

1. Communicate Rollback:
 - Steps:
 - Inform stakeholders (e.g., engineering, product, and data science teams) about the rollback.
 - Provide details on the reasons for the rollback and the steps taken to mitigate the issue.

2. Document Findings:

- Steps:
 - Document the findings from the analysis and troubleshooting.
 - Create a detailed report outlining the issues, the rollback process, and the mitigation steps.

Example Rollback Plan

1. Roll Back to Previous Model:

- Action: Deploy the previous model with one-hot encoding.
- Steps:
 - Replace the current model deployment with the previous one-hot encoding model.
 - Ensure the deployment is done in a controlled manner (e.g., A/B testing, canary release).

2. Monitor Post-Rollback:

- Steps:
 - Monitor the model's performance in the production environment.
 - Ensure that the precision is restored to the baseline and other metrics are stable.

3. Analyze Metrics:

- Steps:
 - Compare the pre- and post-rollback performance metrics.
 - Check the feature distribution in the serving environment.

4. Review Logs and Alerts:

- Steps:
 - Review logs from the rollback deployment.
 - Investigate any errors or warnings that might indicate problems.

5. Examine Feature Influence:

- Steps:
 - Analyze the feature importance and influence in the previous model.
 - Use partial dependence plots to understand the impact of the `plan\_tier` feature.

6. Document Findings:

- Steps:
 - Create a detailed report

Engineering counterpart
7:48 AM - Responded in 0s

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

Agent response

7:48 AM - Responded in 7s

Concrete Next Steps with Owners and Timing

Immediate Rollback

1. Roll Back to Previous Model

- Owner: Model Deployment Team
- Action: Deploy the previous model with one-hot encoding.
- Steps:
 - Replace the current model deployment with the previous one-hot encoding model.
 - Ensure the deployment is done in a controlled manner (e.g., A/B testing, canary release).
- Timeline: within 2 hours

2. Monitor Post-Rollback

- Owner: Monitoring and Alerting Team
- Action: Monitor the model's performance in the production environment.
- Steps:
 - Continuously monitor the model's performance metrics (e.g., precision, AUC) in the production environment.
 - Verify that the precision is restored to the baseline and other metrics are stable.
- Timeline: continuously monitor for 24 hours

Detailed Analysis and Troubleshooting

1. Analyze Metrics

- Owner: Data Science and ML Engineering Teams
- Action: Compare pre- and post-rollback performance metrics.
- Steps:
 - Compare the metrics from the previous version and the current version.
 - Identify any significant differences.
- Timeline: 2-4 hours

2. Review Logs and Alerts

- Owner: Operations and DevOps Teams
- Action: Review logs and any alerts triggered during the rollback.
- Steps:
 - Review logs from the rollback deployment.
 - Investigate any errors or warnings that might indicate problems.
- Timeline: 2-4 hours

3. Examine Feature Influence

- Owner: Data Science and Feature Engineering Teams
- Action: Analyze the feature importance and influence in the previous model.
- Steps:
 - Use techniques like partial dependence plots to understand the impact of the `plan\_tier` feature.
 - Document any insights and findings.
- Timeline: 4-6 hours

Communication and Stakeholder Updates

1. Communicate Rollback

- Owner: Communication and Stakeholder Engagement Team
- Action: Inform stakeholders about the rollback.
- Steps:
 - Provide details on the reasons for the rollback.
 - Explain the steps taken to mitigate the issue.
- Timeline: within 2 hours

2. Document Findings

- Owner: Documentation Team
- Action: Create a detailed report outlining the findings from the analysis and troubleshooting.
- Steps:
 - Document the findings, steps taken, and any recommendations.
 - Share the report with relevant stakeholders.
- Timeline: 6-8 hours

Summary

1. Roll Back to Previous Model: Deploy the previous model with one-hot encoding within 2 hours.
2. Monitor Post-Rollback: Continuously monitor the model's performance for 24 hours.
3. Analyze Metrics: Compare pre- and post-rollback metrics and identify differences within 4 hours.
4. Review Logs and Alerts: Investigate logs and alerts to find any issues within 4 hours.
5. Examine Feature Influence: Analyze feature importance using partial dependence plots within 6 hours.
6. Communicate Rollback: Inform stakeholders within 2 hours.
7. Document Findings: Create a detailed report and share it within 8 hours.

This plan ensures a structured and timely approach to addressing the issue, with clear ownership and timelines for each step.

Engineering counterpart
7:48 AM - Responded in 0s

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

Agent response

7:48 AM - Responded in 7s

The one thing I would want a second pair of eyes on is the feature distribution analysis. Specifically, I would want someone to double-check the feature distribution in the serving environment to ensure that the skew is accurately identified and that the mitigation steps are appropriately targeted.

Detailed Plan

1. Roll Back to Previous Model

- Owner: Model Deployment Team
- Action: Deploy the previous model with one-hot encoding.
- Steps:
 - Replace the current model deployment with the previous one-hot encoding model.
 - Ensure the deployment is done in a controlled manner (e.g., A/B testing, canary release).
- Timeline: within 2 hours

2. Monitor Post-Rollback

- Owner: Monitoring and Alerting Team
- Action: Monitor the model's performance in the production environment.
- Steps:
 - Continuously monitor the model's performance metrics (e.g., precision, AUC) in the production environment.
 - Verify that the precision is restored to the baseline and other metrics are stable.
- Timeline: continuously monitor for 24 hours

3. Feature Distribution Analysis

- Owner: Data Science and Feature Engineering Teams
- Action: Double-check the feature distribution in the serving environment.
- Steps:
 - Verify the distribution of `plan\_tier` in the serving data to ensure it matches the reported skew.
 - Use tools like Feature Store or custom scripts to validate the distribution.
 - Document the findings and any discrepancies.
- Timeline: 2-4 hours

4. Detailed Analysis and Troubleshooting

- Owner: Data Science and ML Engineering Teams
- Action: Compare pre- and post-rollback performance metrics.
- Steps:
 - Compare the metrics from the previous version and the current version.
 - Identify any significant differences.
- Timeline: 4-6 hours

5. Review Logs and Alerts

- Owner: Operations and DevOps Teams
- Action: Review logs and any alerts triggered during the rollback.
- Steps:
 - Review logs from the rollback deployment.
 - Investigate any errors or warnings that might indicate problems.
- Timeline: 4-6 hours

6. Examine Feature Influence

- Owner: Data Science and Feature Engineering Teams
- Action: Analyze the feature importance and influence in the previous model.
- Steps:
 - Use techniques like partial dependence plots to understand the impact of the `plan\_tier` feature.
 - Document any insights and findings.

- Timeline: 6-8 hours

7. Communication and Stakeholder Updates

- Owner: Communication and Stakeholder Engagement Team
- Action: Inform stakeholders about the rollback.
- Steps:
 - Provide details on the reasons for the rollback.
 - Explain the steps taken to mitigate the issue.
- Timeline: within 2 hours

8. Document Findings

- Owner: Documentation Team
- Action: Create a detailed report outlining the findings from the analysis and troubleshooting.
- Steps:
 - Document the findings, steps taken, and any recommendations.
 - Share the report with relevant stakeholders.
- Timeline: 8-10 hours

Summary

1. Roll Back to Previous Model: Deploy the previous model with one-hot encoding within 2 hours.
2. Monitor Post-Rollback: Continuously monitor the model's performance for 24 hours.
3. Feature Distribution Analysis: Double-check the feature distribution in the serving environment to ensure it matches the reported skew.
4. Detailed Analysis and Troubleshooting: Compare pre- and post

Engineering counterpart**7:48 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.