



This PDF is generated from authoritative online content, and is provided for convenience only. This PDF cannot be used for legal purposes. For authoritative understanding of what is and is not supported, always use the online content. To copy code samples, always use the online content.

Predictive Routing Deployment and Operations Guide

Troubleshooting

12/16/2025

Troubleshooting

This topic contains a partial and growing collection of tips and best practices to help you identify and resolve frequently-encountered issues with your Predictive Routing deployment.

Model Training Issues

If model training is taking longer than it should, check the following points:

- Is MongoDB correctly using indexes?
If you see COLLSCAN in the log, it indicates that MongoDB is having trouble with the index set-up. To resolve this issue, re-run one of the queries that resulted in the COLLSCAN message, appending `.explain()` at the end. The result returned will help locate the problem.

Troubleshooting for a URS-based Predictive Routing Environment

If you are using the URS Strategy Subroutines, the following points can help troubleshoot issues with your configuration:

- Does the interaction in question have user data keys with the prefix `prr` attached.
- Does the key `prrResult` have the value `ok`.
- If `prrResult=error`, check the `prrMessage` key for the error message.
- Check if the `prrAgentScore` key contains a valid score. An empty score might be a result of one of the following issues:
 - Normal operation. For example, the agent logged in and received the interaction after it was already scored.
 - The score for the agent is not present in the URS global map. See below for tips on global map troubleshooting.
 - The list of logged-in agents is out of synchronization between the URS and Journey Optimization Platform (JOP).
 - An interaction transfer. Currently, Predictive Routing is supported only on the first leg of the interaction.
- Check if REST API calls to the scoring engine return any errors.
 - Did the strategy authenticate with JOP REST API and receive a token? If not, check whether the request is sent to the correct account in JOP and the correct user name, password, and API key were provided.
 - Was the scoring request formed correctly? Did the call to the subroutine `GetActionFilters` returned a valid skill expression or AgentGroup name?
 - Did the scoring REST API request return the scores for all agents in the target group that URS knows

to be logged in? If not, check whether Agent State Connector lost connection to Stat Server.

- Check if the interaction data was prematurely removed from the URS global map.
 - Did the interaction stay in queue longer than the value set in the **global-map-timeout** option? In this case, the data could have been cleaned automatically.
 - Was the subroutine `PrrlXnCleanup` called at the right time for the interaction?
- Is URS overloaded?
 - Predictive Routing IRD strategy subroutines utilize the `URS TimeBehind[]` function to detect when URS is overloaded and adjust their behavior accordingly. `TimeBehind[]` returns a value that indicates the delay, in milliseconds, between the moment an event is received from T-Server and when it is processed by URS for the current interaction.
 - If the delay is more than 1000 milliseconds, URS is experiencing an overload and is unable to process interactions in a timely manner. In this case, the Predictive Routing IRD subroutines skip agent scoring and matching any new interactions passing through the `ActivatePredictiveRouting` subroutine.
 - If you configured the strategy to hold out for higher-scoring agents, the hold-out is interrupted and interactions are distributed to agents as they become ready, regardless of agent score, until the overload condition ends.
 - When Predictive Routing is deployed in an Orchestration Server strategy and such an overload condition occurs, the `URS SetIdealReadyConditionForORS` subroutine exits through the default port when it is called from the ORS strategy.